




# Malicious Player Card-Based Cryptographic Protocols with a Standard Deck of Cards Using Private Operations

Tomoya Morooka<sup>1</sup>, Yoshifumi Manabe<sup>1</sup>(✉) , and Kazumasa Shinagawa<sup>2,3</sup>

<sup>1</sup> School of Informatics, Kogakuin University, Tokyo, Japan  
manabe@cc.kogakuin.ac.jp

<sup>2</sup> Ibaraki University, 4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan  
kazumasa.shinagawa.np92@vc.ibaraki.ac.jp

<sup>3</sup> National Institute of Advanced Industrial Science and Technology (AIST),  
2-3-26 Aomi, Koto, Tokyo 135-0064, Japan

**Abstract.** This paper shows new card-based cryptographic protocols to compute Boolean functions using a standard deck of cards when the players are malicious. Card-based cryptographic protocols use physical cards instead of computers. They can be used when the software on computers is not reliable. We discuss protocols that use a standard deck of cards because it is easy to prepare. Though protocols that use private operations tend to be efficient in the number of cards used in the protocols, malicious actions are possible during private operations. This paper shows three-player protocols to prevent malicious actions by watching another player's actions. We show logical AND, XOR, and copy protocols since any Boolean functions can be realized by a combination of the protocols. The numbers of cards used by the protocols are the minimum.

**Keywords:** card-based cryptographic protocols · Boolean functions · malicious players · standard deck of cards · multi-party secure computation





## 1 Introduction

Card-based cryptographic protocols [8, 22] were proposed in which physical cards are used instead of computers to securely compute values. They can be used when computers cannot be used or users cannot trust the software on the computer. Also, the protocols are easy to understand, thus the protocols can be used to teach the basics of cryptography [4, 17] to accelerate the social implementation of advanced cryptography [5]. den Boer [2] first showed a five-card protocol to securely compute the logical AND of two inputs. Since then, many protocols have

---

The second author was supported by JSPS KAKENHI Grant Number JP23H00479. The third author was supported during this work by JSPS KAKENHI Grant Numbers JP21K17702 and JP23H00479, and JST CREST Grant Number JPMJCR22M1.

been proposed to realize primitives to compute any Boolean functions [13, 23, 33] and specific computations such as millionaires' problem [16, 24, 28], voting [19, 25, 27, 37], grouping [6], ranking [35], lottery [34], proof of knowledge of a puzzle solution [3, 31], and so on. This paper considers computations of logical AND and logical XOR functions and copy operations since any Boolean function can be realized with a combination of these computations.

Most of the above works are based on a two-color card model. In the two-color card model, there are two kinds of cards,  and . Cards of the same marks cannot be distinguished. In addition, the back of both types of cards is . It is impossible to determine the mark in the back of a given card of . Though the model is simple, such special cards might not be available. Playing cards are easy to prepare, thus protocols using a standard deck of playing cards and their formal security proofs were shown [7, 9, 11, 12, 18, 26, 32]. Recently, protocols that use private operations were shown [14]. Private operations are executed where the other players cannot see, for example, under the table or in the back. The protocols in [14] achieve the minimum number of cards. Though private operations are effective in card-based protocols, there is a problem with private operations. Since the private operations are executed where the other players cannot see, a player might execute malicious actions during private operations. For example, a malicious player might see the marks of face-down cards. Another malicious player might swap the cards to change the values. We need to prevent or detect such malicious actions.

A countermeasure to the problems is watching private actions and detect malicious actions. When the protocols are executed by two players, Alice and Bob, Alice must not see Bob's private actions. If Alice sees Bob's private operations, Alice can see all operations, thus Alice sees the relationship between the private inputs and the output. If the output cards are opened to see the final result, Alice can know the private input data from the relationship. Thus, another player other than the two players is necessary to watch the private operations. If the watcher sees both Alice and Bob's private operations, the watching player can know all operations and the relationship between the input data and the output data. Thus the watching player knows the private data. This paper shows that three players are sufficient to detect malicious actions and keep the protocol secure, just as in the case of the two-color model [29]. In the three-player protocols shown in this paper, Bob watches Alice's private operations, Carol watches Bob's private operations, and Alice watches Carol's private operations.

Few works are done for the case when some players are malicious or make mistakes [1, 10, 15, 20, 21, 36]. [20] discusses information leakage at operation errors. The other works are categorized into two groups. The first one is to use additional cards or special items such as envelopes [10, 15, 21, 36]. The second type introduces the watching player. The watching player for the protocol with a two-color card model is shown [15]. Abe et al. showed a three-player majority voting protocol with a malicious player [1]. Note that the above works are done for the two-color card model. There is no work for a standard deck of cards. As long

as the author knows, this is the first work that discusses malicious activities in protocols that use a standard deck of cards and private operations.

In Sect. 2, basic notations and the private operations introduced in [29] are shown. Section 3 shows logical AND, copy, and logical XOR protocols.

## 2 Preliminaries

### 2.1 Basic Notations

This section gives the notations and basic definitions of card-based protocols with a standard deck of cards. A deck of playing cards consists of 52 distinct mark cards, which are named 1 to 52. The number of each card (for example, 1 is the ace of the spade, and 52 is the king of the club) is common knowledge among the players. The back of all cards is the same  $\boxed{?}$ . It is impossible to determine the mark in the back of a given card of  $\boxed{?}$ .

One-bit data is represented by two cards as follows:  $\boxed{i} \boxed{j} = 0$  and  $\boxed{j} \boxed{i} = 1$  if  $i < j$ .

One pair of cards that represents one bit  $x \in \{0, 1\}$ , whose face is down, is called a commitment of  $x$ , and denoted as  $commit(x)$ . It is written as  $\underbrace{\boxed{?} \boxed{?}}_x$ .

The base of a commitment is the pair of cards used for the commitment. If card  $i$  and  $j$  ( $i < j$ ) are used to set  $commit(x)$ , the commitment is written as  $commit(x)^{\{i,j\}}$  and written as  $\underbrace{\boxed{?} \boxed{?}}_{x^{\{i,j\}}}$ . When the base information is obvious or unnecessary, it is not written.

Note that when these two cards are swapped,  $commit(\bar{x})^{\{i,j\}}$  can be obtained. Thus, logical negation can be computed without private operations.

A set of cards placed in a row is called a sequence of cards. A sequence of cards  $S$  whose length is  $n$  is denoted as  $S = s_1, s_2, \dots, s_n$ , where  $s_i$  is  $i$ -th card of the sequence.  $S = \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{s_1} \dots \underbrace{\boxed{?}}_{s_n}$ . A sequence whose length is even is

called an even sequence.  $S_1 || S_2$  is a concatenation of sequence  $S_1$  and  $S_2$ .

All protocols are executed by three players, Alice, Bob, and Carol. The players are malicious, that is, they might not obey the rule of the protocols and execute any operation. This paper assumes that even malicious players correctly execute misbehavior detection. In the protocols in this paper, a player watches the private operations executed by another player. If a player misbehaves, the watching player detects the malicious action and says that the player misbehaved. The misbehaved player has a punishment for the misbehavior. The detail of the punishment mechanism is out of the scope of this paper. To avoid punishment, malicious players obey the rule of the protocols. Note that the watching player does not output a false misbehavior detection. For the two-color card model, a three-player misbehavior detection protocol without false alarm detection and a four-player misbehavior detection protocol with the ability of false

alarm detection was shown [29]. In order to detect false alarms in a standard deck of cards, four players seem to be necessary. False alarm detection is a further study.

There is no collusion among players, otherwise, private input data can be easily revealed.

The inputs of the protocols are given in a committed format, that is, the players do not know the input values. The output of the protocol must be given in a committed format so that the result can be used as input for further computation.

A protocol is secure when the following two conditions are satisfied: (1) If the output cards are not opened, each player obtains no information about the private input values from the view of the protocol for the player (the sequence of the cards opened to the player). (2) When the output cards are opened, each player obtains no additional information about the private input values other than the information by the output of the protocol. For example, if the output cards of an AND protocol for input  $x$  and  $y$  are opened and the value is 1, the players can know that  $(x, y) = (1, 1)$ . If the output value is 0, the players must not know whether the input  $(x, y)$  is  $(0, 0)$ ,  $(0, 1)$ , or  $(1, 0)$ .

### 2.2 Private Operations

We show three private operations introduced in [29]: private random bisection cuts, private reverse cuts, and private reveals.

**Primitive 1** (*Private random bisection cut*)

A private random bisection cut is the following operation on an even sequence  $S_0 = s_1, s_2, \dots, s_{2m}$ . A player selects a random bit  $b \in \{0, 1\}$  and outputs

$$S_1 = \begin{cases} S_0 & \text{if } b = 0 \\ s_{m+1}, s_{m+2}, \dots, s_{2m}, s_1, s_2, \dots, s_m & \text{if } b = 1 \end{cases}$$

In [29], the operation is executed in a place where the other players cannot see. The player must not disclose the bit  $b$ .

Note that if the private random cut is executed when  $m = 1$  and  $S_0 = \text{commit}(x)^{\{i,j\}}$ , given  $S_0 = \underbrace{\boxed{?} \boxed{?}}_{x^{\{i,j\}}}$ , The player’s output  $S_1 = \underbrace{\boxed{?} \boxed{?}}_{x \oplus b^{\{i,j\}}}$ , which is

$$\underbrace{\boxed{?} \boxed{?}}_{x^{\{i,j\}}} \text{ or } \underbrace{\boxed{?} \boxed{?}}_{\bar{x}^{\{i,j\}}}.$$

Note that a private random bisection cut is the same as the random bisection cut [23], but the operation is not executed in public.

**Primitive 2** (*Private reverse cut*)

A private reverse cut is the following operation on an even sequence  $S_2 = s_1, s_2, \dots, s_{2m}$  and a bit  $b \in \{0, 1\}$ . A player outputs

$$S_3 = \begin{cases} S_2 & \text{if } b = 0 \\ s_{m+1}, s_{m+2}, \dots, s_{2m}, s_1, s_2, \dots, s_m & \text{if } b = 1 \end{cases}$$

In [29], the operation is executed in a place where the other players cannot see. The player must not disclose  $b$ .

Note that the bit  $b$  is not newly selected by the player. This is the difference between the primitive in Primitive 1, where a random bit must be newly selected by the player.

If a player executes a private random bisection cut to  $S$  when the random bit is  $b$  and then executes a private reverse cut using  $b$ , the result is  $S$ .

Note that in some protocols below, selecting left  $m$  cards is executed after a private reverse cut. The sequence of these two operations is called a private reverse selection. A private reverse selection is the following procedure on an even sequence  $S_2 = s_1, s_2, \dots, s_{2m}$  and a bit  $b \in \{0, 1\}$ . A player outputs

$$S_3 = \begin{cases} s_1, s_2, \dots, s_m & \text{if } b = 0 \\ s_{m+1}, s_{m+2}, \dots, s_{2m} & \text{if } b = 1 \end{cases}$$

**Primitive 3** (*Private reveal*). A player privately opens a given committed bit. The player must not disclose the obtained value.

Using the obtained value, the player privately sets a sequence of cards.

Consider the case when Alice executes a private random bisection cut on  $commit(x)$  and Bob executes a private reveal on the bit. Since the committed bit is randomized by the bit  $b$  selected by Alice, the opened bit is  $x \oplus b$ . Even if Bob privately opens the cards, Bob obtains no information about  $x$  if  $b$  is randomly selected and not disclosed by Alice. Bob must not disclose the obtained value. If Bob discloses the obtained value to Alice, Alice knows the value of the committed bit.

### 2.3 Opaque Commitment Pair

An opaque commitment pair is defined as a useful situation for to design a secure protocol using a standard deck of cards [18]. It is a pair of commitments whose bases are unknown to all players. Let us consider the following two commitments using cards  $i, j, i'$ , and  $j'$ . The left (right) commitment has value  $x$  ( $y$ ), respectively, but it is unknown that (1) the left (right) commitment is made using  $i$  and  $j$  ( $i'$  and  $j'$ ), respectively, or (2) the left (right) commitment is made using  $i'$  and  $j'$  ( $i$  and  $j$ ), respectively. Such a pair of commitments is called an opaque commitment pair and written as  $commit(x)^{\{i,j\},\{i',j'\}} || commit(y)^{\{i,j\},\{i',j'\}}$ .

The protocols in this paper use a little different kind of pair, called semi-opaque commitment pair. A player thinks a pair is an opaque commitment pair but another player knows the bases of the commitments. Let us consider the case when a protocol is executed by Alice and Bob. Bob privately makes the pair of commitments with the knowledge of  $x$  and  $y$ . For example, Bob randomly selects a bit  $b \in \{0, 1\}$  and

$$S = \begin{cases} commit(x)^{\{i,j\}} || commit(y)^{\{i',j'\}} & \text{if } b = 0 \\ commit(x)^{\{i',j'\}} || commit(y)^{\{i,j\}} & \text{if } b = 1 \end{cases}$$







then  $S = \text{commit}(x)^{\{i,j\},\{i',j'\}} || \text{commit}(y)^{\{i,j\},\{i',j'\}}$  for Alice. Such a pair is called semi-opaque commitment pair and written as  $\text{commit}(x)^{\{i,j\},\{i',j'\}}|_{\text{Alice}} || \text{commit}(y)^{\{i,j\},\{i',j'\}}|_{\text{Alice}}$ , where the name(s) of the players who think the pair as a opaque commitment pair is written. Note that a name is not written does not mean the player knows the bases of the commitments. For example, the above example says nothing about whether Bob knows the bases or not. Note that the name of the player is written with the initial when it is not ambiguous.

## 2.4 Space and Time Complexities

The space complexity of card-based protocols is evaluated by the number of cards. Minimizing the number of cards is discussed in many works.

The number of rounds was proposed as a criterion to evaluate the time complexity of card-based protocols using private operations [30]. The first round begins from the initial state. In most protocols, a player initially has all cards, but the definition assumes general cases when each player initially has some number of cards. The first round is (possibly parallel) local executions by each player using the cards initially given to each player. It ends at the instant when no further local execution is possible without receiving cards from another player. The local executions in each round include sending cards to some other players but do not include receiving cards.

The  $i (> 1)$ -th round begins with receiving all the cards sent during the  $(i-1)$ -th round. Each player executes local executions using the received cards and the cards left to the player at the end of the  $(i-1)$ -th round. Each player executes local executions until no further local execution is possible without receiving cards from another player. We can define the number of rounds and average rounds. The number of rounds of a protocol is the maximum number of rounds necessary to output the result among all possible inputs and random values. For randomized (Las Vegas) protocols, the average round is the average number of rounds necessary to output the result. Since each operation is relatively simple, the dominating time to execute protocols with private operations is the time to send cards between players and set up so that the cards are not seen by the other players. Thus the number of rounds is the criterion to evaluate the time complexity of card-based protocols with private operations. If the local execution needs many operations, for example,  $O(n)$  operations where  $n$  is the size of the problem, we might need another criterion to consider the cost of local executions.

Let us show an example of a protocol execution, its space complexity, and time complexity with the conventional two-color card model. In the two-color card model, there are two kinds of marks,  and . One-bit data is represented by two cards as follows:  = 0 and  = 1.

**Protocol 1** (*AND protocol in [29]*)

*Input:*  $\text{commit}(x)$  and  $\text{commit}(y)$ .

*Output:*  $\text{commit}(x \wedge y)$ .

1. Alice executes a private random bisection cut on  $\text{commit}(x)$ . Let the output be  $\text{commit}(x')$ . Alice sends  $\text{commit}(x')$  and  $\text{commit}(y)$  to Bob.

2. Bob executes a private reveal on  $\text{commit}(x')$ . Bob privately sets

$$S_2 = \begin{cases} \text{commit}(y) || \text{commit}(0) & \text{if } x' = 1 \\ \text{commit}(0) || \text{commit}(y) & \text{if } x' = 0 \end{cases}$$

and sends  $S_2$  to Alice.

3. Alice executes a private reverse selection on  $S_2$  using the bit  $b$  generated in the private random bisection cut. Let the obtained sequence be  $S_3$ . Alice outputs  $S_3$ .

The AND protocol realizes the following equation.

$$x \wedge y = \begin{cases} y & \text{if } x = 1 \\ 0 & \text{if } x = 0 \end{cases}$$

Our new AND protocol is also based on this equation. The correctness of the protocol is shown in [29]. The number of cards is four since the cards of  $\text{commit}(x')$  are re-used to set  $\text{commit}(0)$ .

Let us consider the time complexity of the protocol. The first round ends at the instant when Alice sends  $\text{commit}(x')$  and  $\text{commit}(y)$  to Bob. The second round begins with receiving the cards by Bob. The second round ends at the instant when Bob sends  $S_2$  to Alice. The third round begins with receiving the cards by Alice. The number of rounds of this protocol is three.

### 3 AND, XOR, and Copy with Three Malicious Players

This section shows our new protocols for AND, XOR, and copy executed by three malicious players. Any malicious action during private operations is detected by a watching player, thus the malicious actions are prohibited if there is no collusion between players.

Bob watches Alice's operations, Carol watches Bob's operations, and Alice watches Carol's operations. All operations are executed in the following manner. Initially, all players are in the same room. If the next operation is executed by Alice, first, Carol exits the room. Then, Alice executes the private operations in front of Bob. Thus, Bob knows all private values. For example, if Alice executes a private random bisection cut, Bob knows the random bit Alice selected. If Alice executes a private reveal, Bob knows the value of the cards Alice opened. If Alice misbehaves, Bob detects the fact and terminates the protocol execution. If there is no misbehavior, Alice's private operations are correctly finished. Then Carol comes back to the room and they execute the next step of the protocol. If the next private operation is executed by Bob(Carol), Alice(Bob) exits from the room, Bob(Carol) executes the private operation in front of Carol(Alice), and Alice(Bob) comes back to the room, respectively.

In the following protocol descriptions, we just write "Alice executes a private operation" to mean "Carol exits the room, Alice executes a private operation in front of Bob, and Carol comes back to the room" for simplicity.

Before we show the protocols, we show a subroutine to fix the base of a given commitment.

### 3.1 Base-Fixed Protocol with Three Players

We show a base-fixed protocol with two inputs  $\text{commit}(x)$  and  $\text{commit}(y)$ . The base of  $\text{commit}(x)$  is fixed to  $\{1, 2\}$ . In the following protocol, the second input value  $y$  is not used as the output, but the value must be kept secret.

The protocol needs private reveals and the values of cards are seen. Before a player sees a value of  $\text{commit}(x)$  and sets cards according to the value, the value must be randomized to hide the value. In the protocol below, Alice sees the value, thus the value must be randomized by the other players. One-player randomization is not enough to hide the private value. Suppose that a player executes a randomization in advance. They obtain  $\text{commit}(x \oplus r)$  and then Alice executes a private reveal. Since Bob watches Alice's execution, Bob knows  $x \oplus r$ . If the randomization  $r$  is executed by Bob, Bob knows  $r$  and  $x \oplus r$  and Bob knows secret value  $x$ . Then consider the case when the randomization is executed by Carol. Alice watches Carol's private operation and knows  $r$ . Since Alice knows  $x \oplus r$  and  $r$ , Alice knows the secret value  $x$ . Therefore, one-player randomization is not enough to hide the private value, and two-player randomizations are necessary. The value must be randomized by Bob and Carol in advance.

Note that the bases of the input commitments are leaked to Alice and Bob during the execution. The protocol can be used only if the information leakage does not cause a security problem, for example, the bases are randomly set by some other player.

**Protocol 2** (*Three player base-fixed protocol*)

*Input:*  $\text{commit}(x)^{\{1,2\},\{3,4\}|A} || \text{commit}(y)^{\{1,2\},\{3,4\}|A}$ .

*Output:*  $\text{commit}(x)^{\{1,2\}}$ .

1. Bob executes a private random bisection cut on each pair using two random bits  $br_1$  and  $br_2$ , respectively. The result  $S_1 = \text{commit}(x \oplus br_1)^{\{1,2\},\{3,4\}|A} || \text{commit}(y \oplus br_2)^{\{1,2\},\{3,4\}|A}$ .
2. Carol executes a private random bisection cut on each pair using two random bits  $cr_1$  and  $cr_2$ , respectively. The result  $S_2 = \text{commit}(x \oplus br_1 \oplus cr_1)^{\{1,2\},\{3,4\}|A} || \text{commit}(y \oplus br_2 \oplus cr_2)^{\{1,2\},\{3,4\}|A}$ .
3. Alice executes a private reveal on both pairs of  $S_2$ . Alice makes  $S_3 = \text{commit}(x \oplus br_1 \oplus cr_1)^{\{1,2\}}$ .
4. Bob executes a private reverse cut using  $br_1$  on  $S_3$ . The result  $S_4 = \text{commit}(x \oplus cr_1)^{\{1,2\}}$ .
5. Carol executes a private reverse cut using  $cr_1$  on  $S_4$ . The result is  $\text{commit}(x)^{\{1,2\}}$ .

**Theorem 1.** *The input values are private in the base-fixed protocol.*

*Proof.* Alice sees  $x \oplus br_1 \oplus cr_1$  and  $y \oplus br_2 \oplus cr_2$  in Step 3. Since Alice watches Carol's private operations, Alice sees  $cr_1$  and  $cr_2$  in Step 2. Alice obtains no information about  $x$  and  $y$  since  $br_1$  and  $br_2$  are unknown to Alice.

Bob knows  $br_1$  and  $br_2$  in Step 1. Since Bob watches Alice's private operations, Bob sees  $x \oplus br_1 \oplus cr_1$  and  $y \oplus br_2 \oplus cr_2$  in Step 3. Bob obtains no information about  $x$  and  $y$  since  $cr_1$  and  $cr_2$  are unknown to Bob.



Carol knows  $cr_1$  and  $cr_2$  in Step 2. Since Carol watches Bob’s private operations, Carol sees  $br_1$  and  $br_2$  in Step 1. Carol obtains no information about  $x$  and  $y$ .  $\square$

### 3.2 AND Protocol

In the following AND, copy, and XOR protocols, the bases of the output commitments are fixed to avoid information leakage from the bases when the outputs are opened.

The outline to execute by three players is as follows. The protocol in [14] has two randomizations. The first is the randomization of the bases of the two input values. The second is the randomization of the input values.

Carol executes private reveals in the following protocol. By the same argument written in the description of the base-fixed protocol, the value must be randomized by the other players in advance. Suppose that Alice and Bob use random bits  $a$  and  $b$  to randomize  $x$ , respectively. After Carol’s private operation using  $x \oplus a \oplus b$ , Alice and Bob execute a private reverse cut using  $a$  and  $b$ , respectively to undo the randomizations. Such randomizations are executed before every private reveals in the protocol.

Next, we need to randomize the bases of the two pairs to hide the relation between the output and inputs. Initially,  $commit(0)$  is made from the cards of  $commit(x)$  using  $\{1, 2\}$  and  $commit(y)$  is made using  $\{3, 4\}$ . Suppose that the output of AND is  $commit(0)$ . It means that  $x = 0$ . If no base change is executed, the base  $\{1, 2\}$  of the output reveal  $x = 0$ . Thus the randomization of bases is necessary. If the base randomization is executed by one player, the private information is known to one player just like the case of randomization of values. Thus the base randomization must be executed by two players.

The detailed protocol is shown below. Note that for the simplicity of description, we write  $S \oplus b$  to mean the pair that the left and the right card are swapped if  $b = 1$ . If  $S = commit(x)$ ,  $S \oplus b$  means  $commit(x \oplus b)$ .

**Protocol 3** (*Three player AND protocol*)

*Input:*  $commit(x)^{\{1,2\}}$  and  $commit(y)^{\{3,4\}}$ .

*Output:*  $commit(x \wedge y)^{\{1,2\}}$ .

1. Alice executes a private random bisection cut on  $commit(x)^{\{1,2\}}$  using random bit  $a_1$ . The result is  $S_1 = commit(x \oplus a_1)^{\{1,2\}}$ .
2. Bob executes a private random bisection cut on  $S_1$  using random bit  $b_1$ . The result is  $S_2 = commit(x \oplus a_1 \oplus b_1)^{\{1,2\}}$ .
3. Carol executes a private reveal on  $S_2$ . Carol sees  $x \oplus a_1 \oplus b_1$ . According to the value, Carol sets  $S_3 || S_4$  as

$$S_3 || S_4 = \begin{cases} commit(0)^{\{1,2\}} || commit(y)^{\{3,4\}} & \text{if } x \oplus a_1 \oplus b_1 = 0 \\ commit(y)^{\{3,4\}} || commit(0)^{\{1,2\}} & \text{if } x \oplus a_1 \oplus b_1 = 1 \end{cases}$$

The cards of  $S_2$  are reused to set  $commit(0)$ .

4. Alice executes a private random bisection cut on  $S_3$  and  $S_4$  using random bit  $a_2$  and  $a_3$ , respectively. The result is  $S_3 \oplus a_2 || S_4 \oplus a_3$ .
5. Bob executes a private random bisection cut on  $S_3 \oplus a_2$  and  $S_4 \oplus a_3$  using random bit  $b_2$  and  $b_3$ , respectively. The result is  $S_3 \oplus a_2 \oplus b_2 || S_4 \oplus a_3 \oplus b_3$ .
6. Carol randomly selects bit  $c_1 \in \{0, 1\}$ . Carol executes private reveals on the two pairs and exchanges the bases of two pairs if  $c_1 = 1$ . Then, Carol executes private random bisection cuts on the two pairs using random bits  $c_2, c_3 \in \{0, 1\}$ . Let the result be  $S_5 || S_6 =$

$$\left\{ \begin{array}{l} \text{commit}(0 \oplus a_2 \oplus b_2 \oplus c_2)^{\{1,2\}} || \text{commit}(y \oplus a_3 \oplus b_3 \oplus c_3)^{\{3,4\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 = 0 \\ \text{commit}(0 \oplus a_2 \oplus b_2 \oplus c_2)^{\{3,4\}} || \text{commit}(y \oplus a_3 \oplus b_3 \oplus c_3)^{\{1,2\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 = 1 \\ \text{commit}(y \oplus a_2 \oplus b_2 \oplus c_2)^{\{3,4\}} || \text{commit}(0 \oplus a_3 \oplus b_3 \oplus c_3)^{\{1,2\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 = 0 \\ \text{commit}(y \oplus a_2 \oplus b_2 \oplus c_2)^{\{1,2\}} || \text{commit}(0 \oplus a_3 \oplus b_3 \oplus c_3)^{\{3,4\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 = 1 \end{array} \right.$$

7. Bob executes private reveals on  $S_5 || S_6$ . Bob randomly selects bit  $b_4 \in \{0, 1\}$ . Bob exchanges the bases of the two commitments if  $b_4 = 1$ . Then Bob executes private reverse cuts on the pairs using  $b_2$  and  $b_3$ , respectively. The result is

$$\left\{ \begin{array}{l} \text{commit}(0 \oplus a_2 \oplus c_2)^{\{1,2\}} || \text{commit}(y \oplus a_3 \oplus c_3)^{\{3,4\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 0 \\ \text{commit}(0 \oplus a_2 \oplus c_2)^{\{3,4\}} || \text{commit}(y \oplus a_3 \oplus c_3)^{\{1,2\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y \oplus a_2 \oplus c_2)^{\{1,2\}} || \text{commit}(0 \oplus a_3 \oplus c_3)^{\{3,4\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y \oplus a_2 \oplus c_2)^{\{3,4\}} || \text{commit}(0 \oplus a_3 \oplus c_3)^{\{1,2\}} \\ \quad \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 0 \end{array} \right.$$

8. Carol executes private reverse cuts on the pairs using  $c_2$  and  $c_3$ , respectively.
9. Alice executes a private reverse cut on each of the pairs using  $a_2$  and  $a_3$ , respectively.

Let  $S_7 || S_8$  be the result after the two private reverse cuts.  $S_7 || S_8 =$

$$\left\{ \begin{array}{ll} \text{commit}(0)^{\{1,2\}} || \text{commit}(y)^{\{3,4\}} & \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 0 \\ \text{commit}(0)^{\{3,4\}} || \text{commit}(y)^{\{1,2\}} & \text{if } x \oplus a_1 \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{1,2\}} || \text{commit}(0)^{\{3,4\}} & \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{3,4\}} || \text{commit}(0)^{\{1,2\}} & \text{if } x \oplus a_1 \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 0 \end{array} \right.$$

Alice then executes a private reverse cut using  $a_1$ . The result is

$$\left\{ \begin{array}{ll} \text{commit}(0)^{\{1,2\}} || \text{commit}(y)^{\{3,4\}} & \text{if } x \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 0 \\ \text{commit}(0)^{\{3,4\}} || \text{commit}(y)^{\{1,2\}} & \text{if } x \oplus b_1 = 0 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{1,2\}} || \text{commit}(0)^{\{3,4\}} & \text{if } x \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{3,4\}} || \text{commit}(0)^{\{1,2\}} & \text{if } x \oplus b_1 = 1 \text{ and } c_1 \oplus b_4 = 0 \end{array} \right.$$

10. Bob executes a private reverse selection using  $b_1$ . Let  $T_0$  be the result and  $T_1$  be the pair that is not selected.

$$T_0 = \begin{cases} \text{commit}(0)^{\{1,2\}} & \text{if } x = 0 \text{ and } c_1 \oplus b_4 = 0 \\ \text{commit}(0)^{\{3,4\}} & \text{if } x = 0 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{1,2\}} & \text{if } x = 1 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(y)^{\{3,4\}} & \text{if } x = 1 \text{ and } c_1 \oplus b_4 = 0 \end{cases}$$

The value of  $T_0$  is  $\text{commit}(x \wedge y)$  and its base is randomly set by  $c_1 \oplus b_4$ . Since Alice does not know  $b_4$ ,  $T_0 = \text{commit}(x \wedge y)^{\{1,2\},\{3,4\}|A}$ .

Similarly,

$$T_1 = \begin{cases} \text{commit}(y)^{\{3,4\}} & \text{if } x = 0 \text{ and } c_1 \oplus b_4 = 0 \\ \text{commit}(y)^{\{1,2\}} & \text{if } x = 0 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(0)^{\{3,4\}} & \text{if } x = 1 \text{ and } c_1 \oplus b_4 = 1 \\ \text{commit}(0)^{\{1,2\}} & \text{if } x = 1 \text{ and } c_1 \oplus b_4 = 0 \end{cases}$$

The value of  $T_1$  is  $\text{commit}(\bar{x} \wedge y)$  and its base is randomly set by  $c_1 \oplus b_4$ .  $T_1 = \text{commit}(\bar{x} \wedge y)^{\{1,2\},\{3,4\}|A}$ .

Next, execute the base-fixed protocol on these pairs. Then the players obtain  $\text{commit}(x \wedge y)^{\{1,2\}}$ .

The protocol is 14 rounds since the first step of the base-fixed protocol is executed by Bob. The semi-honest two-player AND protocol [14] is 8 rounds. The number of cards is four. Since four cards are necessary to input  $x$  and  $y$ , the number of cards is the minimum. The correctness of the output value is shown in the protocol, thus we show the security.

**Theorem 2.** *The AND protocol is secure.*

*Proof.* First, we show the security for Bob. Since Bob watches Alice, Bob knows the values in Steps 1, 2, 4, 5, 7, 9, 10 and Steps 1, 3, and 4 of the base-fixed protocol. Bob thus sees  $a_i, b_i, br_i, x \wedge y \oplus br_1 \oplus cr_1, \bar{x} \wedge y \oplus br_2 \oplus cr_2$ , and  $((0 \oplus a_2 \oplus b_2 \oplus c_2 \text{ and } y \oplus a_3 \oplus b_3 \oplus c_3 \text{ if } x \oplus a_1 \oplus b_1 = 0) \text{ or } (y \oplus a_2 \oplus b_2 \oplus c_2 \text{ and } y \oplus a_3 \oplus b_3 \oplus c_3 \text{ if } x \oplus a_1 \oplus b_1 = 1))$ . Bob can obtain no information about the secret input and output values since the values of cards are randomized by  $c_2, c_3, cr_1$ , or  $cr_2$  that are unknown to Bob.

From the bases of the cards, Bob obtains no information since the bases of two randomized values,  $0 \oplus a_2 \oplus b_2 \oplus c_2$  and  $y \oplus a_3 \oplus b_3 \oplus c_3$  (or  $y \oplus a_2 \oplus b_2 \oplus c_2$  and  $0 \oplus a_3 \oplus b_3 \oplus c_3$ ) are randomized by unknown value  $c_1$ . The bases of two randomized values,  $x \wedge y \oplus br_1 \oplus cr_1$  and  $\bar{x} \wedge y \oplus br_2 \oplus cr_2$  are randomized by  $c_1 \oplus b_4$  but  $c_1$  is unknown to Bob.

Next, we show the security for Carol. Since Carol watches Bob, Carol knows the values in Steps 2, 3, 5, 6, 7, 8, 10 and Steps 1, 2, 4, and 5 of the base-fixed protocol. Carol thus sees  $b_i, c_i, br_i, cr_i, x \oplus a_1 \oplus b_1, 0 \oplus a_2 \oplus b_2, y \oplus a_3 \oplus b_3$ , and  $((0 \oplus a_2 \oplus b_2 \oplus c_2 \text{ and } y \oplus a_3 \oplus b_3 \oplus c_3 \text{ if } x \oplus a_1 \oplus b_1 = 0) \text{ or } (y \oplus a_2 \oplus b_2 \oplus c_2 \text{ and } y \oplus a_3 \oplus b_3 \oplus c_3 \text{ if } x \oplus a_1 \oplus b_1 = 1))$ . From the cards, Carol obtains no

information about the secret input values since the values are randomized by unknown values  $a_1$ ,  $a_2$ , or  $a_3$ .

About the bases of the cards, Carol knows whether she set  $\text{commit}(0)^{\{1,2\}} || \text{commit}(y)^{\{3,4\}}$  or  $\text{commit}(0)^{\{3,4\}} || \text{commit}(y)^{\{1,2\}}$  in Step 3 and both two base randomizations by Carol and Bob, thus she knows whether  $S_7 || S_8$  is  $\text{commit}(0) || \text{commit}(y)$  or  $\text{commit}(y) || \text{commit}(0)$  and each commitment is made by  $\{1, 2\}$  or  $\{3, 4\}$ . However, Carol cannot see the private reverse cut by Alice in Step 9, Carol cannot know which pair is selected as the final result thus no information is known to Carol. Since Alice sets the base to  $\{1, 2\}$ , Carol cannot know information about the secret input values from the base of the final result.

Last, we show the security for Alice. Alice knows the values in Steps 1, 3, 4, 6, 8, 9, and Steps 2, 3, and 5 of the base-fixed protocol. Alice thus sees  $a_i$ ,  $c_i$ ,  $cr_i$ ,  $x \oplus a_1 \oplus b_1$ ,  $x \wedge y \oplus br_1 \oplus cr_1$ , and  $\bar{x} \wedge y \oplus br_2 \oplus cr_2$ , and  $((0 \oplus a_2 \oplus b_2 \oplus c_2$  and  $y \oplus a_3 \oplus b_3 \oplus c_3$  if  $x \oplus a_1 \oplus b_1 = 0$ ) or  $(y \oplus a_2 \oplus b_2 \oplus c_2$  and  $y \oplus a_3 \oplus b_3 \oplus c_3$  if  $x \oplus a_1 \oplus b_1 = 1$ ). From the revealed cards, Alice obtains no information about the secret input and output values since each value is randomized by unknown value  $b_1$ ,  $b_2$ ,  $b_3$ ,  $br_1$ , or  $br_2$ .

Alice knows whether  $S_3 || S_4$  is  $\text{commit}(0)^{\{1,2\}} || \text{commit}(y)^{\{3,4\}}$  or  $\text{commit}(y)^{\{3,4\}} || \text{commit}(0)^{\{1,2\}}$ . Alice also knows the bases of each pair of  $S_5 || S_6$ . Though Alice knows the bases of  $S_5 || S_6$ , Bob's base change using  $b_4$  is unknown to Alice. Thus, the bases of  $T_0$  and  $T_1$  are random for Alice because of  $b_4$ . When Alice sees  $x \wedge y \oplus br_1 \oplus cr_1$  and  $\bar{x} \wedge y \oplus br_2 \oplus cr_2$  in Step 3 of the base-fixed protocol, the bases are randomized by  $c_1 \oplus b_4$ . Thus, Alice obtains no information from the bases of the commitments.  $\square$

### 3.3 Copy Protocol

Next, we show a new copy protocol by three players.

#### Protocol 4 (Three player copy protocol)

*Input:*  $\text{commit}(x)^{\{1,2\}}$  and two new cards 3 and 4.

*Output:*  $\text{commit}(x)^{\{1,2\}}$  and  $\text{commit}(x)^{\{3,4\}}$

1. Alice executes a private random bisection cut on  $\text{commit}(x)^{\{1,2\}}$  using random bit  $a$ . The result is  $\text{commit}(x \oplus a)^{\{1,2\}}$ .
2. Bob executes a private random bisection cut on  $\text{commit}(x \oplus a)^{\{1,2\}}$  using random bit  $b$ . The result is  $\text{commit}(x \oplus a \oplus b)^{\{1,2\}}$ .
3. Carol executes a private reveal on  $\text{commit}(x \oplus a \oplus b)^{\{1,2\}}$  and sees  $x \oplus a \oplus b$ . Carol privately makes  $\text{commit}(x \oplus a \oplus b)^{\{3,4\}}$ .
4. Alice executes a private reverse cut on each of the pairs using  $a$ . The result is  $\text{commit}(x \oplus b)^{\{1,2\}}$  and  $\text{commit}(x \oplus b)^{\{3,4\}}$ .
5. Bob executes a private reverse cut on each of the pairs using  $b$ . The result is  $\text{commit}(x)^{\{1,2\}}$  and  $\text{commit}(x)^{\{3,4\}}$ .

The number of cards is the minimum. The protocol is five rounds. The semi-honest two-player copy protocol [14] is three rounds.

**Theorem 3.** *The copy protocol is secure.*

*Proof.* Alice sees  $a$  and  $x \oplus a \oplus b$ . Bob sees  $a$  and  $b$ . Carol sees  $b$  and  $x \oplus a \oplus b$ . Thus no player knows the secret value  $x$ .  $\square$

### 3.4 XOR Protocol

Since AND and copy protocols are shown and NOT is obvious, any Boolean function can be realized by the combination of these protocols. XOR protocol is shown because the realization of XOR is simple.

**Protocol 5** (*Three player XOR protocol*)

*Input:*  $\text{commit}(x)^{\{1,2\}}$  and  $\text{commit}(y)^{\{3,4\}}$ .

*Output:*  $\text{commit}(x \oplus y)^{\{1,2\}}$ .

1. Alice executes a private random bisection cut on  $\text{commit}(x)^{\{1,2\}}$  and  $\text{commit}(y)^{\{3,4\}}$  using the same random bit  $a \in \{0,1\}$ . The result is  $\text{commit}(x \oplus a)^{\{1,2\}}$  and  $\text{commit}(y \oplus a)^{\{3,4\}}$ .
2. Bob executes a private random bisection cut on  $\text{commit}(x \oplus a)^{\{1,2\}}$  and  $\text{commit}(y \oplus a)^{\{3,4\}}$  using the same random bit  $b \in \{0,1\}$ . The result is  $\text{commit}(x \oplus a \oplus b)^{\{1,2\}}$  and  $\text{commit}(y \oplus a \oplus b)^{\{3,4\}}$ .
3. Carol executes a private reveal on  $\text{commit}(y \oplus a \oplus b)^{\{3,4\}}$ . Carol sees  $y \oplus a \oplus b$ . Carol executes a private reverse cut on  $\text{commit}(x \oplus a \oplus b)^{\{1,2\}}$  using  $y \oplus a \oplus b$ . The result is  $\text{commit}((x \oplus a \oplus b) \oplus (y \oplus a \oplus b))^{\{1,2\}} = \text{commit}(x \oplus y)^{\{1,2\}}$ .

The protocol is three rounds. The semi-honest two-player XOR protocol [14] is two rounds. The protocol uses four cards. Since any protocol needs four cards to input  $x$  and  $y$ , the number of cards is the minimum.

**Theorem 4.** *The XOR protocol is secure.*

*Proof.* Alice sees  $a$  and  $y \oplus a \oplus b$ . Bob sees  $a$  and  $b$ . Carol sees  $b$  and  $y \oplus a \oplus b$ . Thus no player knows the secret value  $y$ .  $\square$

## References

1. Abe, Y., Iwamoto, M., Ohta, K.: How to detect malicious behaviors in a card-based majority voting protocol with three inputs. In: 2020 International Symposium on Information Theory and Its Applications (ISITA), pp. 377–381. IEEE (2020)
2. den Boer, B.: More efficient match-making and satisfiability *the five card trick*. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 208–217. Springer, Heidelberg (1990). [https://doi.org/10.1007/3-540-46885-4\\_23](https://doi.org/10.1007/3-540-46885-4_23)
3. Bultel, X., et al.: Physical zero-knowledge proof for Makaro. In: Izumi, T., Kuznetsov, P. (eds.) SSS 2018. LNCS, vol. 11201, pp. 111–125. Springer, Cham (2018). [https://doi.org/10.1007/978-3-030-03232-6\\_8](https://doi.org/10.1007/978-3-030-03232-6_8)
4. Cheung, E., Hawthorne, C., Lee, P.: CS 758 project: secure computation with playing cards (2013). [http://cdchawthorne.com/writings/secure\\_playing\\_cards.pdf](http://cdchawthorne.com/writings/secure_playing_cards.pdf)

5. Hanaoka, G., et al.: Physical and visual cryptography to accelerate social implementation of advanced cryptographic technologies. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* (2023). (In Japanese)
6. Hashimoto, Y., Shinagawa, K., Nuida, K., Inamura, M., Hanaoka, G.: Secure grouping protocol using a deck of cards. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **101**(9), 1512–1524 (2018)
7. Koch, A.: Cryptographic protocols from physical assumptions. Ph.D. thesis, Karlsruhe Institute of Technology, Germany (2019)
8. Koch, A.: The landscape of optimal card-based protocols. *Math. Cryptol.* **1**(2), 115–131 (2021)
9. Koch, A., Schrempf, M., Kirsten, M.: Card-based cryptography meets formal verification. *N. Gener. Comput.* **39**(1), 115–158 (2021)
10. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. In: *Proceedings of 10th International Conference on Fun with Algorithms (FUN 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
11. Koyama, H., Miyahara, D., Mizuki, T., Sone, H.: A secure three-input and protocol with a standard deck of minimal cards. In: Santhanam, R., Musatov, D. (eds.) *CSR 2021*. LNCS, vol. 12730, pp. 242–256. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-79416-3\\_14](https://doi.org/10.1007/978-3-030-79416-3_14)
12. Koyama, H., Toyoda, K., Miyahara, D., Mizuki, T.: New card-based copy protocols using only random cuts. In: *Proceedings of the 8th ACM on ASIA Public-Key Cryptography Workshop, APKC 2021*, pp. 13–22. Association for Computing Machinery, New York (2021)
13. Manabe, Y.: Survey: card-based cryptographic protocols to calculate primitives of Boolean functions. *Int. J. Comput. Softw. Eng.* **27**(1), 178 (2022)
14. Manabe, Y., Ono, H.: Card-based cryptographic protocols with a standard deck of cards using private operations. In: Cerone, A., Ölveczky, P.C. (eds.) *ICTAC 2021*. LNCS, vol. 12819, pp. 256–274. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-85315-0\\_15](https://doi.org/10.1007/978-3-030-85315-0_15)
15. Manabe, Y., Ono, H.: Card-based cryptographic protocols with malicious players using private operations. *N. Gener. Comput.* **40**(1), 67–93 (2022)
16. Miyahara, D., Hayashi, Y.I., Mizuki, T., Sone, H.: Practical card-based implementations of Yao’s millionaire protocol. *Theoret. Comput. Sci.* **803**, 207–221 (2020)
17. Mizuki, T.: Applications of card-based cryptography to education. In: *IEICE Technical Report ISEC2016-53*, pp. 13–17 (2016). (In Japanese)
18. Mizuki, T.: Efficient and secure multiparty computations using a standard deck of playing cards. In: Foresti, S., Persiano, G. (eds.) *CANS 2016*. LNCS, vol. 10052, pp. 484–499. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-48965-0\\_29](https://doi.org/10.1007/978-3-319-48965-0_29)
19. Mizuki, T., Asiedu, I.K., Sone, H.: Voting with a logarithmic number of cards. In: Mauri, G., Dennunzio, A., Manzoni, L., Porreca, A.E. (eds.) *UCNC 2013*. LNCS, vol. 7956, pp. 162–173. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-39074-6\\_16](https://doi.org/10.1007/978-3-642-39074-6_16)
20. Mizuki, T., Komano, Y.: Information leakage due to operative errors in card-based protocols. *Inf. Comput.* **285**, 104910 (2022)
21. Mizuki, T., Shizuya, H.: Practical card-based cryptography. In: Ferro, A., Luccio, F., Widmayer, P. (eds.) *FUN 2014*. LNCS, vol. 8496, pp. 313–324. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-07890-8\\_27](https://doi.org/10.1007/978-3-319-07890-8_27)
22. Mizuki, T., Shizuya, H.: Computational model of card-based cryptographic protocols and its applications. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **100**(1), 3–11 (2017)

23. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) FAW 2009. LNCS, vol. 5598, pp. 358–369. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-02270-8\\_36](https://doi.org/10.1007/978-3-642-02270-8_36)
24. Nakai, T., Misawa, Y., Tokushige, Y., Iwamoto, M., Ohta, K.: How to solve millionaires' problem with two kinds of cards. *N. Gener. Comput.* **39**(1), 73–96 (2021)
25. Nakai, T., Shirouchi, S., Iwamoto, M., Ohta, K.: Four cards are sufficient for a card-based three-input voting protocol utilizing private permutations. In: Shikata, J. (ed.) ICITS 2017. LNCS, vol. 10681, pp. 153–165. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-72089-0\\_9](https://doi.org/10.1007/978-3-319-72089-0_9)
26. Niemi, V., Renvall, A.: Solitaire zero-knowledge. *Fundam. Inform.* **38**(1, 2), 181–188 (1999)
27. Nishida, T., Mizuki, T., Sone, H.: Securely computing the three-input majority function with eight cards. In: Dediu, A.-H., Martín-Vide, C., Truthe, B., Vega-Rodríguez, M.A. (eds.) TPNC 2013. LNCS, vol. 8273, pp. 193–204. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-45008-2\\_16](https://doi.org/10.1007/978-3-642-45008-2_16)
28. Ono, H., Manabe, Y.: Efficient card-based cryptographic protocols for the millionaires' problem using private input operations. In: Proceedings of 13th Asia Joint Conference on Information Security (AsiaJCIS 2018), pp. 23–28 (2018)
29. Ono, H., Manabe, Y.: Card-based cryptographic logical computations using private operations. *N. Gener. Comput.* **39**(1), 19–40 (2021)
30. Ono, H., Manabe, Y.: Minimum round card-based cryptographic protocols using private operations. *Cryptography* **5**(3), 17 (2021)
31. Sasaki, T., Miyahara, D., Mizuki, T., Sone, H.: Efficient card-based zero-knowledge proof for sudoku. *Theoret. Comput. Sci.* **839**, 135–142 (2020)
32. Shinagawa, K., Mizuki, T.: Secure computation of any Boolean function based on any deck of cards. In: Chen, Y., Deng, X., Lu, M. (eds.) FAW 2019. LNCS, vol. 11458, pp. 63–75. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-18126-0\\_6](https://doi.org/10.1007/978-3-030-18126-0_6)
33. Shinagawa, K., Nuida, K.: A single shuffle is enough for secure card-based computation of any Boolean circuit. *Discret. Appl. Math.* **289**, 248–261 (2021)
34. Shinoda, Y., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based covert lottery. In: Maimut, D., Oprina, A.-G., Sauveron, D. (eds.) SecITC 2020. LNCS, vol. 12596, pp. 257–270. Springer, Cham (2021). [https://doi.org/10.1007/978-3-030-69255-1\\_17](https://doi.org/10.1007/978-3-030-69255-1_17)
35. Takashima, K., et al.: Card-based protocols for secure ranking computations. *Theoret. Comput. Sci.* **845**, 122–135 (2020)
36. Takashima, K., Miyahara, D., Mizuki, T., Sone, H.: Actively revealing card attack on card-based protocols. *Nat. Comput.* **21**(4), 615–628 (2022)
37. Watanabe, Y., Kuroki, Y., Suzuki, S., Koga, Y., Iwamoto, M., Ohta, K.: Card-based majority voting protocols with three inputs using three cards. In: Proceedings of 2018 International Symposium on Information Theory and Its Applications (ISITA), pp. 218–222. IEEE (2018)