

Card-based Cryptographic Protocols with a Standard Deck of Cards Using Private Operations

Yoshifumi Manabe¹[0000–0002–6312–257X] and Hibiki Ono¹

Kogakuin University, Shinjuku, Tokyo 163–8677 Japan.
manabe@cc.kogakuin.ac.jp





Abstract. This paper shows new kinds of card-based cryptographic protocols with a standard deck of cards using private operations. They are multi-party secure computations executed by multiple players without computers. Most card-based cryptographic protocols use a special deck of cards that consists of many cards with two kinds of marks. Though these protocols are simple and efficient, the users need to prepare such special cards. Few protocols were shown that use a standard deck of playing cards. Though the protocols with a standard deck of cards can be easily executed in our daily life, the numbers of cards used by these protocols are larger than the ones that use the special deck of cards. This paper shows logical AND, logical XOR, and copy protocols for a standard deck of cards that use the minimum number of cards. Any Boolean functions can be calculated with a combination of the above protocols. The new protocols use private operations that are executed by a player where the other players cannot see. The results show the effectiveness of private operations in card-based cryptographic protocols.

Keywords: Multi-party secure computation · card-based cryptographic protocols · private operations · logical computations · copy · playing cards.

1 Introduction

Card-based cryptographic protocols [13, 34, 36] were proposed in which physical cards are used instead of computers to securely calculate 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, 28]. den Boer [2] first showed a five-card protocol to securely calculate logical AND of two inputs. Since then, many protocols have been proposed to realize primitives to calculate any Boolean functions [7, 12, 17, 37, 48, 57] and specific computations such as a class of Boolean functions [1, 23, 24, 29, 33, 42, 43, 45, 50, 51, 55, 62, 64], millionaires' problem [25, 39, 46], realizing Turing machines [6, 15], voting [31, 40, 44, 63], random permutation [8, 10, 11, 38], grouping [9], ranking [60], lottery [58], proof of knowledge of a puzzle solution [3, 5, 21, 26, 27, 49, 52–54], and so on. This paper considers calculations of

logical AND and logical XOR functions and copy operation since any Boolean function can be realized with a combination of these calculations.

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 are not available in our daily life. When the players make the special cards using white cards and a printer, the person who prints the marks to cards might add tiny marks on the cards for the person to distinguish the cards and obtain secret data. Thus, hand-made cards are not so easy to realize.

To solve the problem, card-based cryptographic protocols using a standard deck of playing cards were shown [14, 18, 19, 30, 41, 56]. Playing cards are available at many houses and easy to buy. Niemi and Renvall first showed protocols that use a standard deck of playing cards [41]. They showed logical XOR, logical AND, and copy protocols since any Boolean functions can be realized by a combination of these protocols. Their protocols are ‘Las Vegas’ type protocols, that is, the execution times of the protocols are not limited. The protocols are expected to terminate within a finite time, but if the sequence of the random numbers is bad, the protocols do not terminate forever. Mizuki showed fixed time logical XOR, logical AND, and copy protocols [30]. Though the number of cards used by the XOR protocol is the minimum, the ones used by the logical AND and copy protocols are not the minimum. Koch et al. showed a four-card ‘Las Vegas’ type AND protocol and it is impossible to obtain four-card finite time protocol with the model without private operations [14]. Koyama et al. showed a three-input ‘Las Vegas’ type AND protocol with the minimum number of cards [18]. Koyama et al. showed an efficient ‘Las Vegas’ type copy protocol [19]. Shinagawa and Mizuki showed protocols to calculate any n -variable function using a standard deck of playing cards and a deck of UNO¹ cards [56].

Randomization or a private operation is the most important primitive in these card-based protocols. If every primitive executed in a card-based protocol is deterministic and public, the relationship between the private input values and output values is known to the players. When the output value is disclosed, the private input value can be known to the players from the relationship. Thus, all protocols need some random or private operation.

First, public randomization primitives have been discussed and then recently, private operations are considered. Many protocols use random bisection cuts [37], which randomly execute swapping two decks of cards or not swapping. If the random value used in the randomization is disclosed, the secret input value is known to the players. If some player privately brings a high-speed camera, the random value selected by the randomization might be known by analyzing the image. Though the size of a high-speed camera is very large, the size might become very small shortly. To prepare for the situation, we need to consider using private operations.

¹ <https://www.letsplayuno.com>

Operations that a player executes in a place where the other players cannot see are called private operations. These operations are considered to be executed under the table or in the back. Private operations are shown to be the most powerful primitives in card-based cryptographic protocols. They were first introduced to solve millionaires' problem [39]. Using three private operations shown later, committed-input and committed-output logical AND, logical XOR, and copy protocols can be achieved with the minimum number of cards on the two-color card model [48]. Another class of private operations is private input operations that are used when a player inputs a private value [20, 46, 59, 63]. These operations are not discussed in this paper since the protocols need the players to know the input values. The protocols without private input operations can be used when the players do not know private input values.

So the research question is whether we can achieve the minimum number of cards for a standard deck of cards if we use private operations. We show positive results to the question. This paper shows new logical AND and copy protocols with a standard deck of playing cards that achieves the minimum number of cards by using private operations. The results show that the private operations are also effective for a standard deck of cards.

Note that in this paper, all players are assumed to be semi-honest. Few works are done for the case when some players are malicious or make mistakes [16, 22, 32, 35, 61].

In Section 2, basic notations and the private operations introduced in [48] are shown. Section 3 shows logical AND, copy, and logical XOR protocols. Then, protocols to calculate any n -variable Boolean function are shown. Section 4 concludes the paper.

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 as 1 to 52. The number of each card (for example, 1 is the ace of spade and 52 is the king of 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 $\text{commit}(x)$. 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 $\text{commit}(x)$ (That is, set $\boxed{i}\boxed{j}$ if $x = 0$ and set

$\begin{array}{|c|c|} \hline \mathbf{j} & \mathbf{i} \\ \hline \end{array}$ if $x = 1$), the commitment is written as $\text{commit}(x)^{\{i,j\}}$ and written as $\underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \end{array}}_{x^{\{i,j\}}}$. When the base information is obvious or unnecessary, it is not written.

Note that when these two cards are swapped, $\text{commit}(\bar{x})^{\{i,j\}}$ can be obtained. Thus, logical negation can be calculated 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. $S = \underbrace{\begin{array}{|c|} \hline ? \\ \hline \end{array}}_{s_1} \underbrace{\begin{array}{|c|} \hline ? \\ \hline \end{array}}_{s_2} \underbrace{\begin{array}{|c|} \hline ? \\ \hline \end{array}}_{s_3} \dots \underbrace{\begin{array}{|c|} \hline ? \\ \hline \end{array}}_{s_n}$. 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 two players, Alice and Bob. The players are semi-honest, that is, they obey the rule of the protocols but try to obtain secret values. There is no collusion between Alice and Bob, 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 an input to further calculation.

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 $x = 1$ and $y = 1$. If the output value is 0, the players must not know whether the input (x, y) is $(0, 0)$, $(0, 1)$, or $(1, 0)$.

The following protocols use random numbers. Random numbers can be generated without computers using coin-flipping or some similar methods. During the protocol executions, cards are sent and received between the players. The communication is executed by handing the cards between the players to avoid information leakage during the communication. If the players are not in the same place during the protocol execution, a trusted third party (for example, post office) is necessary to send and receive cards between players.

2.2 Private operations

We show three private operations introduced in [48]: 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

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

The player executes this operation 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 $S_0 = \text{commit}(x)$, given $S_0 = \underbrace{\boxed{?} \boxed{?}}_x$, The player's output $S_1 = \underbrace{\boxed{?} \boxed{?}}_{x \oplus b}$, which is $\underbrace{\boxed{?} \boxed{?}}_x$ or $\underbrace{\boxed{?} \boxed{?}}_{\bar{x}}$.

Note that a private random bisection cut is the same as the random bisection cut [37], but the operation is executed in a hidden place.

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

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

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

The player executes this operation 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.

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 $\text{commit}(x)$ 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 to design a secure protocol using a standard deck of cards [30]. It is a pair of commitments whose

bases are unknown to a player. 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 pair of commitment is called an opaque commitment pair and written as $\text{commit}(x)^{\{i,j\},\{i',j'\}} || \text{commit}(y)^{\{i,j\},\{i',j'\}}$. Note that there is a case when Alice thinks a pair is an opaque commitment pair but Bob knows the base, especially when 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







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

then $S = \text{commit}(x)^{\{i,j\},\{i',j'\}} || \text{commit}(y)^{\{i,j\},\{i',j'\}}$ for Alice.

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 [47]. The first round begins from the initial state. 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 result of every private execution is known to the player. For example, shuffling whose result is unknown to the player himself is not executed. Since the private operations are executed in a place where the other players cannot see, it is hard to force the player to execute such operations whose result is unknown to the player. 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. 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.

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 [48])

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.

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

The correctness of the protocol is shown in [48]. 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 at receiving the cards by Bob. The second round ends at the instant when Bob sends S_2 to Alice. The third round begins at receiving the cards by Alice. The number of rounds of this protocol is three.

Since each operation is relatively simple, the dominating time to execute protocols with private operations is the time to sending cards between players and setting 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.

2.5 Problems with a standard deck of cards

The above AND protocol cannot be executed as it is with a standard deck of cards.

The protocol uses the property that all \heartsuit cards (\clubsuit cards) are indistinguishable. Even if the final cards are opened to see the result, it is impossible to know that the opened cards are the cards of $\text{commit}(y)$ or $\text{commit}(0)$. If it is possible to detect the above information, the value of x is known to the players.

First, let us consider a simple encoding using a standard deck of a playing card that heart and diamond cards mean \heartsuit and all club and spade cards mean \clubsuit . With this simple encoding, let us consider the case when the aces of diamond and spade are used to set $\text{commit}(x)$ and the aces of heart and club are used to set $\text{commit}(y)$.

Suppose that $x = 1$ and $y = 0$. In this case, the result is $\text{commit}(y)$, thus the result is correct since $y = 0$. At step 2 of the protocol, aces of diamond and spade are re-used to set $\text{commit}(0)$. Since $x = 1$, the result is $\text{commit}(y)$. When the cards are opened to see the result, the cards are the aces of heart and club.

The players can know that y is selected as the output, thus x must be 1. The execution reveals the information of inputs from the cards used to set the input commitments.

Next, consider the case when the encoding rule $\boxed{i}\boxed{j} = 0, \boxed{j}\boxed{i} = 1$ if $i < j$ is used to the standard deck of playing cards. Suppose again that $x = 1$ and $y = 0$. When two inputs are given as $\text{commit}(x)^{\{1,2\}}$ and $\text{commit}(y)^{\{3,4\}}$, $\text{commit}(0)$ and $\text{commit}(y)$ are set as $\text{commit}(0)^{\{1,2\}}$ and $\text{commit}(y)^{\{3,4\}}$, respectively at Step 2. Since $x = 1$, the result is $\text{commit}(y)^{\{3,4\}}$. When the cards are opened to see the result, the cards are 3 and 4. The players can know that y is selected as the output, thus x must be 1. This execution also reveals the information of inputs from the base of the commitments.

When we design a protocol with a standard deck of cards, we must consider the information leakage from the base of the commitment.

3 AND, XOR, and copy with a standard deck of cards

This section shows our new protocols for AND, and copy with the minimum number of cards using private operations. We also show XOR protocol using private operations to show the minimum number of cards can also be achieved using private operations. Before we show the protocols, we show subroutines to change the base of a given commitment.

3.1 Base Change Protocols

A base change protocol changes the base of a commitment without changing the value of the commitment. A base change protocol is also shown in [30], but the protocol uses a public shuffle, thus we show a new protocol that uses private operations.

Protocol 2 (*Base change protocol (1)*)

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

Output: $\text{commit}(x)^{\{3,4\}}$.

1. Bob executes a private random bisection cut on $\text{commit}(x)^{\{1,2\}}$. Let $b \in \{0, 1\}$ be the bit Bob selected. The result is $S_1 = \text{commit}(x \oplus b)^{\{1,2\}}$. Bob sends S_1 to Alice.
2. Alice executes a private reveal on S_1 . Alice sees $x \oplus b$. Alice makes $S_2 = \text{commit}(x \oplus b)^{\{3,4\}}$ and sends S_2 to Bob.
3. Bob executes a private reverse cut using b on S_2 . The result is $\text{commit}(x)^{\{3,4\}}$.

The protocol is three rounds. The security of the protocol is as follows. When Alice sees the cards at Step 2, the value is $x \oplus b$. Since b is a random value unknown to Alice, Alice has no information about x by the reveal. Bob sees no open cards, thus Bob has no information about x . Note again that Bob must not disclose b to Alice.

Another base change protocol from an opaque commitment pair can be considered. In the following protocol, the second input value \perp is random and meaningless to Alice.

Protocol 3 (*Base change protocol (2)*)

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

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

1. Bob executes a private random bisection cut on the left pair, $\text{commit}(x)^{\{1,2\},\{3,4\}}$. Let $b \in \{0,1\}$ be the bit Bob selected. The result $S_1 = \text{commit}(x \oplus b)^{\{1,2\},\{3,4\}} || \text{commit}(\perp)^{\{1,2\},\{3,4\}}$. Bob sends S_1 to Alice.
2. Alice executes a private reveal on S_1 . Alice sees $x \oplus b$. If the base of the left pair is $\{1,2\}$, Alice just faces down the left pair and the cards, S_2 , be the result. Otherwise, the base of the right pair is $\{1,2\}$. Alice makes $S_2 = \text{commit}(x \oplus b)^{\{1,2\}}$ using the right cards. Alice sends S_2 to Bob.
3. Bob executes a private reverse cut using b on S_2 . The result is $\text{commit}(x)^{\{1,2\}}$.

In this protocol, Alice knows the bases of the input commitments. The protocol can be used only when this information leakage does not cause a security problem, for example, the bases are randomly set by Bob. The security of the input value x is just the same as the first base change protocol.

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.

Protocol 4 (*AND protocol*)

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

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

1. Alice executes a private random bisection cut on $\text{commit}(x)^{\{1,2\}}$ and $\text{commit}(y)^{\{3,4\}}$ using two different bits b_1 and b_2 . Alice sends the results, $S_1 = \text{commit}(x \oplus b_1)^{\{1,2\}}$ and $S_2 = \text{commit}(y \oplus b_2)^{\{3,4\}}$, to Bob.
2. Bob executes private reveals on S_1 and S_2 . Bob sees $x \oplus b_1$ and $y \oplus b_2$. Bob randomly selects bit $b_3 \in \{0,1\}$. Bob privately sets

$$S_{3,0} = \begin{cases} \text{commit}(x \oplus b_1)^{\{1,2\}} & \text{if } b_3 = 0 \\ \text{commit}(x \oplus b_1)^{\{3,4\}} & \text{if } b_3 = 1 \end{cases}$$

and

$$S_{3,1} = \begin{cases} \text{commit}(y \oplus b_2)^{\{3,4\}} & \text{if } b_3 = 0 \\ \text{commit}(y \oplus b_2)^{\{1,2\}} & \text{if } b_3 = 1 \end{cases}$$

$S_{3,0} = \text{commit}(x \oplus b_1)^{\{1,2\},\{3,4\}}$ and $S_{3,1} = \text{commit}(y \oplus b_2)^{\{1,2\},\{3,4\}}$ for Alice. Bob sends $S_{3,1}$ to Alice.

3. Alice executes a private reverse cut using b_2 on $S_{3,1}$. The result $S'_{3,1} = \text{commit}(y)^{\{1,2\},\{3,4\}}$. Alice sends $S'_{3,1}$ to Bob.
4. Bob executes a private reveal on $S_{3,0}$ and sees $x \oplus b_1$. Bob privately sets cards

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

Note that the cards used for $S_{3,0}$ are reused to set $\text{commit}(0)$. Since $S_{3,0} = \text{commit}(\cdot)^{\{1,2\},\{3,4\}}$, the result is $\text{commit}(0)^{\{1,2\},\{3,4\}}$ for Alice. Bob sends S_4 to Alice.

5. Alice executes a private reverse selection on S_4 using b_1 . Let S_5 be the result and the remaining two cards be S_6 . The result $S_5 = \text{commit}(y)^{\{1,2\},\{3,4\}}$ if $(b_1 = 0 \text{ and } x \oplus b_1 = 1) \text{ or } (b_1 = 1 \text{ and } x \oplus b_1 = 0)$. The condition equals to $x = 1$.
 $S_5 = \text{commit}(0)^{\{1,2\},\{3,4\}}$ if $(b_1 = 0 \text{ and } x \oplus b_1 = 0) \text{ or } (b_1 = 1 \text{ and } x \oplus b_1 = 1)$. The condition equals to $x = 0$. Thus,

$$\begin{aligned} S_5 &= \begin{cases} \text{commit}(y)^{\{1,2\},\{3,4\}} & \text{if } x = 1 \\ \text{commit}(0)^{\{1,2\},\{3,4\}} & \text{if } x = 0 \end{cases} \\ &= \text{commit}(x \wedge y)^{\{1,2\},\{3,4\}} \end{aligned}$$

Alice sends S_5 and S_6 to Bob.

6. Bob executes a private random bisection cut on S_6 to erase the value to Alice. Let b' be the random bit selected by Bob and S'_6 be the result. Bob and Alice execute Protocol 3 (Base change protocol (2)) to $S_5 || S'_6$. Then they obtain $\text{commit}(x \wedge y)^{\{1,2\}}$.

The protocol is eight rounds since the first round of the base change protocol can be executed in the sixth round of AND protocol by Bob. 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 1. *The AND protocol is secure.*

Proof. First, we show the security for Bob. Though Bob sees cards at Step 2 and 4, the cards, $S_1 = \text{commit}(x \oplus b_1)^{\{1,2\}}$ and $S_2 = \text{commit}(y \oplus b_2)^{\{3,4\}}$, are randomized by b_1 and b_2 . Thus Bob obtains no information about the input values.

Alice sees cards at the second step of the base change protocol. At Step 3 after the private reverse selection by Alice,

$$S'_{3,1} = \begin{cases} \text{commit}(y)^{\{3,4\}} & \text{if } b_3 = 0 \\ \text{commit}(y)^{\{1,2\}} & \text{if } b_3 = 1 \end{cases}$$

and $\text{commit}(y)$ ($\text{commit}(0)$) is finally selected as S_5 if $x = 1$ ($x = 0$), respectively. The value is then randomized using b as $\text{commit}(y \oplus b)$ ($\text{commit}(b)$) at Step 1

of the base change protocol (2).

$$S_6 = \begin{cases} \text{commit}(0)^{\{1,2\},\{3,4\}} & \text{if } x = 1 \\ \text{commit}(y)^{\{1,2\},\{3,4\}} & \text{if } x = 0 \end{cases}$$

S_6 is also randomized at Step 6 using b' .

Thus at Step 2 of the base change protocol (2), Alice sees the randomized cards of $S_5 || S_6$, which are

$$\begin{cases} \text{commit}(b)^{\{1,2\}} || \text{commit}(y \oplus b')^{\{3,4\}} & \text{if } b_3 = 0 \text{ and } x = 0 \\ \text{commit}(y \oplus b)^{\{3,4\}} || \text{commit}(b')^{\{1,2\}} & \text{if } b_3 = 0 \text{ and } x = 1 \\ \text{commit}(b)^{\{3,4\}} || \text{commit}(y \oplus b')^{\{1,2\}} & \text{if } b_3 = 1 \text{ and } x = 0 \\ \text{commit}(y \oplus b)^{\{1,2\}} || \text{commit}(b')^{\{3,4\}} & \text{if } b_3 = 1 \text{ and } x = 1 \end{cases}$$

Therefore, Alice sees

$$\begin{cases} \text{commit}(0)^{\{1,2\}} || \text{commit}(0)^{\{3,4\}} & \text{if } (b_3 = 0 \wedge x = 0 \wedge b = 0 \wedge y \oplus b' = 0) \vee (b_3 = 1 \wedge x = 1 \wedge y \oplus b = 0 \wedge b' = 0) \\ \text{commit}(0)^{\{1,2\}} || \text{commit}(1)^{\{3,4\}} & \text{if } (b_3 = 0 \wedge x = 0 \wedge b = 0 \wedge y \oplus b' = 1) \vee (b_3 = 1 \wedge x = 1 \wedge y \oplus b = 0 \wedge b' = 1) \\ \text{commit}(1)^{\{1,2\}} || \text{commit}(0)^{\{3,4\}} & \text{if } (b_3 = 0 \wedge x = 0 \wedge b = 1 \wedge y \oplus b' = 0) \vee (b_3 = 1 \wedge x = 1 \wedge y \oplus b = 1 \wedge b' = 0) \\ \text{commit}(1)^{\{1,2\}} || \text{commit}(1)^{\{3,4\}} & \text{if } (b_3 = 0 \wedge x = 0 \wedge b = 1 \wedge y \oplus b' = 1) \vee (b_3 = 1 \wedge x = 1 \wedge y \oplus b = 1 \wedge b' = 1) \\ \text{commit}(0)^{\{3,4\}} || \text{commit}(0)^{\{1,2\}} & \text{if } (b_3 = 0 \wedge x = 1 \wedge y \oplus b = 0 \wedge b' = 0) \vee (b_3 = 1 \wedge x = 0 \wedge b = 0 \wedge y \oplus b' = 0) \\ \text{commit}(0)^{\{3,4\}} || \text{commit}(1)^{\{1,2\}} & \text{if } (b_3 = 0 \wedge x = 1 \wedge y \oplus b = 0 \wedge b' = 1) \vee (b_3 = 1 \wedge x = 0 \wedge b = 0 \wedge y \oplus b' = 1) \\ \text{commit}(1)^{\{3,4\}} || \text{commit}(0)^{\{1,2\}} & \text{if } (b_3 = 0 \wedge x = 1 \wedge y \oplus b = 1 \wedge b' = 0) \vee (b_3 = 1 \wedge x = 0 \wedge b = 1 \wedge y \oplus b' = 0) \\ \text{commit}(1)^{\{3,4\}} || \text{commit}(1)^{\{1,2\}} & \text{if } (b_3 = 0 \wedge x = 1 \wedge y \oplus b = 1 \wedge b' = 1) \vee (b_3 = 1 \wedge x = 0 \wedge b = 1 \wedge y \oplus b' = 1) \end{cases}$$

Let P_{ij} ($i \in \{0, 1\}, j \in \{0, 1\}$) be the probability when $x = i$ and $y = j$. The probabilities $P(b = 0), P(b = 1), P(b' = 0), P(b_3 = 0)$, and $P(b_3 = 1)$ are $1/2$, thus the probabilities when Alice sees $\text{commit}(v)^{\{i,i+1\}} || \text{commit}(w)^{\{4-i,5-i\}}$ ($v, w \in \{0, 1\}, i \in \{1, 3\}$) are the same value $(P_{00} + P_{01} + P_{10} + P_{11})/8$. Thus, Alice obtains no information from the cards she sees. \square

The comparison of AND protocols is shown in Table 1.

3.3 Copy protocol

Next, we show a new copy protocol. Note that the protocol is essentially the same as the one in [48] for the two-color card model. The number of cards is the minimum.

Table 1. Comparison of AND protocols with a standard deck of cards.

Article	# of cards	Note
Niemi et al. [41]	5	Las Vegas algorithm
Koch et al. [14]	4	Las Vegas algorithm
Mizuki [30]	8	Fixed time algorithm
This paper	4	Fixed time algorithm

Table 2. Comparison of copy protocols with a standard deck of cards

Article	# of cards	Note
Niemi et al. [41]	6	Las Vegas algorithm
Koyama et al. [19]	6	Las Vegas algorithm
Mizuki [30]	6	Fixed time algorithm
This paper	4	Fixed time algorithm

Protocol 5 (*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 $\text{commit}(x)^{\{1,2\}}$. Let b the random bit Alice selects. Alice sends the result, $\text{commit}(x \oplus b)^{\{1,2\}}$, to Bob.
2. Bob executes a private reveal on $\text{commit}(x \oplus b)^{\{1,2\}}$ and sees $x \oplus b$. Bob privately makes $\text{commit}(x \oplus b)^{\{3,4\}}$. Bob sends $\text{commit}(x \oplus b)^{\{1,2\}}$ and $\text{commit}(x \oplus b)^{\{3,4\}}$ to Alice.
3. Alice 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 protocol is three rounds.

Theorem 2. *The copy protocol is secure.*

Proof. Since Alice sees no open cards, Alice obtains no information about the input value. Though Bob sees $x \oplus b$, input x is randomized by b and Bob obtains no information about x . \square

The comparison of copy protocols are shown in Table 2.

The number of rounds can be decreased to two if we use six cards using the protocol in [47] for the two-color card model.

3.4 XOR protocol

Though the minimum number of cards is already achieved in [30], the protocol uses public shuffles. We show a new protocol that uses private operations. The protocol is essentially the same as the one in [47] for the two-color card model.

Protocol 6 (*XOR protocol*)

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

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

Table 3. Comparison of XOR protocols with a standard deck of cards.

Article	# of cards	Note
Niemi et al. [41]	4	Las Vegas algorithm
Mizuki [30]	4	Fixed time algorithm
This paper	4	Fixed time algorithm

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 $b \in \{0, 1\}$. The result is $\text{commit}(x \oplus b)^{\{1,2\}}$ and $\text{commit}(y \oplus b)^{\{3,4\}}$. Alice sends these cards to Bob.
2. Bob executes a private reveal on $\text{commit}(y \oplus b)^{\{3,4\}}$. Bob sees $y \oplus b$. Bob executes a private reverse cut on $\text{commit}(x \oplus b)^{\{1,2\}}$ using $y \oplus b$. The result is $\text{commit}((x \oplus b) \oplus (y \oplus b))^{\{1,2\}} = \text{commit}(x \oplus y)^{\{1,2\}}$.

The protocol 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.

Note that if Bob sends $\text{commit}(y \oplus b)^{\{3,4\}}$ to Alice and Alice executes a private reverse cut using b , an input $\text{commit}(y)^{\{3,4\}}$ can be obtained without additional cards. This protocol is called an input preserving XOR and it is used in Section 3.5.

Theorem 3. *The XOR protocol is secure.*

Proof. Since Alice sees no open cards, Alice obtains no information about the input values. Though Bob sees $y \oplus b$, input y is randomized by b and Bob obtains no information about y . \square

The comparison of XOR protocols is shown in Table 3.

3.5 Any Boolean function

We show two kinds of protocols to calculate any n -variable Boolean function. The first one uses many cards but the number of rounds is constant. The second one uses fewer cards but needs many rounds. Let $f(x_1, x_2, \dots, x_n)$ be an n -variable Boolean function.

Protocol 7 (Protocol for any n -variable Boolean function (1))

Input: $\text{commit}(x_i)^{\{2i-1, 2i\}} (i = 1, 2, \dots, n)$.

Output: $\text{commit}(f(x_1, x_2, \dots, x_n))^{\{1, 2\}}$.

1. Alice executes a private random bisection cut on $\text{commit}(x_i)^{\{2i-1, 2i\}} (i = 1, 2, \dots, n)$. Let the output be $\text{commit}(x'_i)^{\{2i-1, 2i\}} (i = 1, 2, \dots, n)$. Note that one random bit b_i is selected for each $x_i (i = 1, 2, \dots, n)$. $x'_i = x_i \oplus b_i (i = 1, 2, \dots, n)$. Alice sends $\text{commit}(x'_i)^{\{2i-1, 2i\}} (i = 1, 2, \dots, n)$ to Bob.

2. Bob executes a private reveal on $\text{commit}(x'_i)^{\{2i-1, 2i\}}$ ($i = 1, 2, \dots, n$). Bob selects a random bit $b \in \{0, 1\}$. Bob privately makes 2^n commitments S_{a_1, a_2, \dots, a_n} ($a_i \in \{0, 1\}, i = 1, 2, \dots, n$) as $S_{a_1, a_2, \dots, a_n} = \text{commit}(f(a_1 \oplus x'_1, a_2 \oplus x'_2, \dots, a_n \oplus x'_n) \oplus b)$ using card $3, 4, \dots, 2^{n+1} + 1, 2^{n+1} + 2$. Note that the cards used to set each commitment are randomly selected by Bob. Bob executes a private random bisection cut on $\text{commit}(\cdot)^{\{1, 2\}}$ to erase the value. Bob sends these commitments to Alice.
3. Alice privately reveals S_{b_1, b_2, \dots, b_n} . Alice sees $f(b_1 \oplus x'_1, b_2 \oplus x'_2, \dots, b_n \oplus x'_n) \oplus b = f(x_1, x_2, \dots, x_n) \oplus b$, since $x'_i = x_i \oplus b_i$ ($i = 1, 2, \dots, n$). Alice privately makes $S = \text{commit}(f(x_1, x_2, \dots, x_n) \oplus b)^{\{1, 2\}}$ and sends S to Bob.
4. Bob executes a private reverse cut using b on S . The result is $\text{commit}(f(x_1, x_2, \dots, x_n))^{\{1, 2\}}$. Bob outputs the result.

Note that Bob can re-use cards of $3, 4, \dots, 2n - 1$, and $2n$ to set S_{a_1, a_2, \dots, a_n} . The protocol uses $2^{n+1} + 2$ cards. The number of rounds is four.

Theorem 4. *The protocol 7 is secure.*

Proof. Bob sees $x'_i = x_i \oplus b_i$, but the input x_i is randomized by b_i and Bob obtains no information about x_i . Alice sees $f(x_1, x_2, \dots, x_n) \oplus b$, but the value is randomized by b and Alice obtains no information about $f(x_1, x_2, \dots, x_n)$. Alice obtains no information from the base of the commitment since the base is randomly selected by Bob. \square

The main idea of the other protocol is the same as the one in [48] for the two-color card model, which uses an input preserving AND protocol. After the AND protocol, the unused pair of cards has $g = \bar{x} \wedge y$ [48]. Let $h = x \wedge y$. The last step of AND protocol (the first step of the base change protocol) is changed so that Alice sets $\text{commit}(h \oplus b)^{\{1, 2\}}$ and $\text{commit}(g \oplus b')^{\{3, 4\}}$. By the private reverse cut by Bob, Bob obtains $\text{commit}(h)^{\{1, 2\}}$ and $\text{commit}(g)^{\{3, 4\}}$. Execute the input preserving XOR protocol to g and h so that h is preserved. The output $g \oplus h = x \wedge y \oplus \bar{x} \wedge y = y$, thus we can obtain $\text{commit}(x \wedge y)^{\{1, 2\}}$ and $\text{commit}(y)^{\{3, 4\}}$. Therefore, one input can be preserved without additional cards by the AND protocol.

Any Boolean function $f(x_1, x_2, \dots, x_n)$ can be represented as follows:

$$f(x_1, x_2, \dots, x_n) = \bar{x}_1 \wedge \bar{x}_2 \wedge \dots \wedge \bar{x}_n \wedge f(0, 0, \dots, 0) \oplus x_1 \wedge \bar{x}_2 \wedge \dots \wedge \bar{x}_n \wedge f(1, 0, \dots, 0) \oplus \dots \oplus x_1 \wedge x_2 \wedge \dots \wedge x_n \wedge f(1, 1, \dots, 1).$$

Since the terms with $f(i_1, i_2, \dots, i_n) = 0$ can be removed, this function f can be written as $f = \bigoplus_{i=1}^k v_1^i \wedge v_2^i \wedge \dots \wedge v_n^i$, where $v_j^i = x_j$ or \bar{x}_j . Let us write $T_i = v_1^i \wedge v_2^i \wedge \dots \wedge v_n^i$. The number of terms $k (< 2^n)$ depends on f .

Protocol 8 (Protocol for any n -variable Boolean function (2))

Input: $\text{commit}(x_i)^{\{2i+3, 2i+4\}}$ ($i = 1, 2, \dots, n$).

Output: $\text{commit}(f(x_1, x_2, \dots, x_n))^{\{1, 2\}}$.

The additional four cards (two pairs of cards) 1, 2, 3, and 4 are used as follows. 1 and 2 store the intermediate value to calculate f .

Table 4. Comparison of protocols to calculate any n -variable Boolean function with a standard deck of cards.

Article	# of cards	Note
Shinagawa et al. [56]	$2n + 8$	Fixed time algorithm
This paper's Protocol 7	$2^{n+1} + 2$	Fixed time algorithm
This paper's Protocol 8	$2n + 4$	Fixed time algorithm

3 and 4 store the intermediate value to calculate T_i .

Execute the following steps for $i = 1, 2, \dots, k$.

1. Copy v_1^i from the input $\text{commit}(x_1)$ as $\text{commit}(v_1^i)^{\{3,4\}}$. (Note that if v_1^i is \bar{x}_1 , NOT is taken after the copy).
2. For $j = 2, \dots, n$, execute the following procedure: Execute the input preserving AND protocol to $\text{commit}(\cdot)^{\{3,4\}}$ and $\text{commit}(v_j^i)$ so that input $\text{commit}(v_j^i)$ is preserved. The result is stored as $\text{commit}(\cdot)^{\{3,4\}}$. (Note that if v_j^i is \bar{x}_j , NOT is taken before the AND protocol and NOT is taken again for the preserved input.)
At the end of this step, T_i is obtained as $\text{commit}(v_1^i \wedge v_2^i \wedge \dots \wedge v_n^i)^{\{3,4\}}$.
3. If $i = 1$, copy $\text{commit}(\cdot)^{\{3,4\}}$ to $\text{commit}(\cdot)^{\{1,2\}}$. If $i > 1$, apply the XOR protocol between $\text{commit}(\cdot)^{\{3,4\}}$ and $\text{commit}(\cdot)^{\{1,2\}}$. The result is stored as $\text{commit}(\cdot)^{\{1,2\}}$.

At the end of the protocol, $\text{commit}(f(x_1, x_2, \dots, x_n))^{\{1,2\}}$ is obtained.

The comparison of protocols to calculate any n -variable Boolean function is shown in Table 4.

The number of additional cards in [56] with a standard deck of cards is 8. Thus the number of additional cards is reduced using private operations.

4 Conclusion

This paper showed AND, XOR, and copy protocols that use a standard deck of cards. The numbers of cards used by the protocols are the minimum. The results show the effectiveness of private operations. One of the remaining problems is obtaining protocols when a player is malicious.

Acknowledgements The authors would like to thank anonymous referees for their careful reading of our manuscript and their many insightful comments and suggestions.

References

1. Abe, Y., Hayashi, Y.i., Mizuki, T., Sone, H.: Five-card and computations in committed format using only uniform cyclic shuffles. New Generation Computing pp. 1–18 (2021)

2. den Boer, B.: More efficient match-making and satisfiability the five card trick. In: Proc. of EUROCRYPT '89, LNCS Vol. 434. pp. 208–217 (1990)
3. Bultel, X., Dreier, J., Dumas, J.G., Lafourcade, P., Miyahara, D., Mizuki, T., Nagao, A., Sasaki, T., Shinagawa, K., Sone, H.: Physical zero-knowledge proof for makaro. In: Proc. of 20th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS 2018), LNCS Vol.11201. pp. 111–125 (2018)
4. Cheung, E., Hawthorne, C., Lee, P.: Cs 758 project: Secure computation with playing cards (2013), http://cdhawthorne.com/writings/secure_playing_cards.pdf
5. Dumas, J.G., Lafourcade, P., Miyahara, D., Mizuki, T., Sasaki, T., Sone, H.: Interactive physical zero-knowledge proof for norinori. In: Proc. of 25th International Computing and Combinatorics Conference(COCOON 2019), LNCS Vol. 11653. pp. 166–177. Springer (2019)
6. Dvořák, P., Koucký, M.: Barrington plays cards: The complexity of card-based protocols. arXiv preprint arXiv:2010.08445 (2020)
7. Francis, D., Aljunid, S.R., Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Necessary and sufficient numbers of cards for securely computing two-bit output functions. In: Proc. of Second International Conference on Cryptology and Malicious Security(Mycrypt 2016), LNCS Vol. 10311. pp. 193–211 (2017)
8. Hashimoto, Y., Nuida, K., Shinagawa, K., Inamura, M., Hanaoka, G.: Toward finite-runtime card-based protocol for generating hidden random permutation without fixed points. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **101-A**(9), 1503–1511 (2018)
9. Hashimoto, Y., Shinagawa, K., Nuida, K., Inamura, M., Hanaoka, G.: Secure grouping protocol using a deck of cards. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **101**(9), 1512–1524 (2018)
10. Ibaraki, T., Manabe, Y.: A more efficient card-based protocol for generating a random permutation without fixed points. In: Proc. of 3rd Int. Conf. on Mathematics and Computers in Sciences and in Industry (MCSI 2016). pp. 252–257 (2016)
11. Ishikawa, R., Chida, E., Mizuki, T.: Efficient card-based protocols for generating a hidden random permutation without fixed points. In: Proc. of 14th International Conference on Unconventional Computation and Natural Computation(UCNC 2015), LNCS Vol. 9252. pp. 215–226 (2015)
12. Kastner, J., Koch, A., Walzer, S., Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: The minimum number of cards in practical card-based protocols. In: Proc. of Asiacrypt 2017, Part III, LNCS Vol. 10626. pp. 126–155 (2017)
13. Koch, A.: The landscape of optimal card-based protocols. IACR Cryptology ePrint Archive, Report 2018/951 (2018)
14. Koch, A., Schrempf, M., Kirsten, M.: Card-based cryptography meets formal verification. New Generation Computing **39**(1), 115–158 (2021)
15. Koch, A., Walzer, S.: Private function evaluation with cards. Cryptology ePrint Archive, Report 2018/1113 (2018), <https://eprint.iacr.org/2018/1113>
16. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. In: Proc. of 10th International Conference on Fun with Algorithms (FUN 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
17. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Proc. of Asiacrypt 2015, LNCS Vol. 9452. pp. 783–807 (2015)
18. 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.) Proc.

- of 16th International Computer Science Symposium in Russia (CSR 2021), LNCS Vol. 12730. pp. 242–256. Springer International Publishing, Cham (2021)
19. 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. pp. 13–22. APKC ‘21, Association for Computing Machinery, New York, NY, USA (2021). <https://doi.org/10.1145/3457338.3458297>
 20. Kurosawa, K., Shinozaki, T.: Compact card protocol. In: Proc. of 2017 Symposium on Cryptography and Information Security(SCIS 2017). pp. 1A2–6 (2017), (In Japanese)
 21. Lafourcade, P., Miyahara, D., Mizuki, T., Sasaki, T., Sone, H.: A physical zkp for slitherlink: How to perform physical topology-preserving computation. In: Proc. of 15th International Conference on Information Security Practice and Experience(ISPEC 2019), LNCS Vol. 11879. pp. 135–151. Springer (2019)
 22. Manabe, Y., Ono, H.: Secure card-based cryptographic protocols using private operations against malicious players. In: Proc. of 13th International Conference on Information Technology and Communications Security(SecITC 2020), LNCS Vol. 12596. pp. 55–70. Springer (2020)
 23. Manabe, Y., Ono, H.: Card-based cryptographic protocols for three-input functions using private operations. In: Proc. of 32nd International Workshop on Combinatorial Algorithms (IWOCA 2021), LNCS Vol. 12757. Springer (2021), to appear
 24. Marcedone, A., Wen, Z., Shi, E.: Secure dating with four or fewer cards. IACR Cryptology ePrint Archive, Report 2015/1031 (2015)
 25. Miyahara, D., Hayashi, Y.i., Mizuki, T., Sone, H.: Practical card-based implementations of yao’s millionaire protocol. Theoretical Computer Science **803**, 207–221 (2020)
 26. Miyahara, D., Robert, L., Lafourcade, P., Takeshige, S., Mizuki, T., Shinagawa, K., Nagao, A., Sone, H.: Card-based zkp protocols for takuzu and juosan. In: Proc. of 10th International Conference on Fun with Algorithms (FUN 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
 27. Miyahara, D., Sasaki, T., Mizuki, T., Sone, H.: Card-based physical zero-knowledge proof for kakuro. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **102**(9), 1072–1078 (2019)
 28. Mizuki, T.: Applications of card-based cryptography to education. In: IEICE Technical Report ISEC2016-53. pp. 13–17 (2016), (In Japanese)
 29. Mizuki, T.: Card-based protocols for securely computing the conjunction of multiple variables. Theoretical Computer Science **622**, 34–44 (2016)
 30. Mizuki, T.: Efficient and secure multiparty computations using a standard deck of playing cards. In: Proc. of 15th International Conference on Cryptology and Network Security(CANS 2016), LNCS Vol.10052. pp. 484–499. Springer (2016)
 31. Mizuki, T., Asiedu, I.K., Sone, H.: Voting with a logarithmic number of cards. In: Proc. of 12th International Conference on Unconventional Computing and Natural Computation (UCNC 2013), LNCS Vol. 7956. pp. 162–173 (2013)
 32. Mizuki, T., Komano, Y.: Analysis of information leakage due to operative errors in card-based protocols. In: Proc. of 29th International Workshop on Combinatorial Algorithms(IWOCA 2019), LNCS Vol. 10979. pp. 250–262. Springer (2018)
 33. Mizuki, T., Kumamoto, M., Sone, H.: The five-card trick can be done with four cards. In: Proc. of Asiacrypt 2012, LNCS Vol.7658. pp. 598–606 (2012)
 34. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. International Journal of Information Security **13**(1), 15–23 (2014)

35. Mizuki, T., Shizuya, H.: Practical card-based cryptography. In: Proc. of 7th International Conference on Fun with Algorithms(FUN2014), LNCS Vol. 8496. pp. 313–324 (2014)
36. Mizuki, T., Shizuya, H.: Computational model of card-based cryptographic protocols and its applications. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **100**(1), 3–11 (2017)
37. Mizuki, T., Sone, H.: Six-card secure and and four-card secure xor. In: Proc. of 3rd International Workshop on Frontiers in Algorithms(FAW 2009), LNCS Vol. 5598. pp. 358–369 (2009)
38. Murata, S., Miyahara, D., Mizuki, T., Sone, H.: Efficient generation of a card-based uniformly distributed random derangement. In: Proc. of 15th International Workshop on Algorithms and Computation (WALCOM 2021), LNCS Vol. 12635. pp. 78–89. Springer International Publishing, Cham (2021)
39. Nakai, T., Misawa, Y., Tokushige, Y., Iwamoto, M., Ohta, K.: How to solve millionaires’ problem with two kinds of cards. New Generation Computing **39**(1), 73–96 (2021)
40. Nakai, T., Shirouchi, S., Iwamoto, M., Ohta, K.: Four cards are sufficient for a card-based three-input voting protocol utilizing private sends. In: Proc. of 10th International Conference on Information Theoretic Security (ICITS 2017), LNCS Vol. 10681. pp. 153–165 (2017)
41. Niemi, V., Renvall, A.: Solitaire zero-knowledge. Fundamenta Informaticae **38**(1, 2), 181–188 (1999)
42. Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols for any boolean function. In: Proc. of 15th International Conference on Theory and Applications of Models of Computation(TAMC 2015), LNCS Vol. 9076. pp. 110–121 (2015)
43. Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Securely computing three-input functions with eight cards. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **98**(6), 1145–1152 (2015)
44. Nishida, T., Mizuki, T., Sone, H.: Securely computing the three-input majority function with eight cards. In: Proc. of 2nd International Conference on Theory and Practice of Natural Computing(TPNC 2013), LNCS Vol. 8273. pp. 193–204 (2013)
45. Nishimura, A., Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols using unequal division shuffles. Soft Computing **22**(2), 361–371 (2018)
46. Ono, H., Manabe, Y.: Efficient card-based cryptographic protocols for the millionaires’ problem using private input operations. In: Proc. of 13th Asia Joint Conference on Information Security(AsiaJCIS 2018). pp. 23–28 (2018)
47. Ono, H., Manabe, Y.: Card-based cryptographic protocols with the minimum number of rounds using private operations. In: Proc. of 14th International Workshop on Data Privacy Management (DPM 2019) LNCS Vol. 11737. pp. 156–173 (2019)
48. Ono, H., Manabe, Y.: Card-based cryptographic logical computations using private operations. New Generation Computing **39**(1), 19–40 (2021)
49. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Interactive physical zkp for connectivity:applications to nurikabe and hitori. In: Proc. of 17th International Conference on Computability in Europe(CiE 2021), LNCS (2021)
50. Ruangwises, S., Itoh, T.: And protocols using only uniform shuffles. In: Proc. of 14th International Computer Science Symposium in Russia(CSR 2019), LNCS Vol. 11532. pp. 349–358 (2019)

51. Ruangwises, S., Itoh, T.: Securely computing the n -variable equality function with $2n$ cards. In: Proc. of 16th International Conference on Theory and Applications of Models of Computation (TAMC 2020), LNCS Vol. 12337. pp. 25–36. Springer (2020)
52. Ruangwises, S., Itoh, T.: Physical zero-knowledge proof for numberlink puzzle and k vertex-disjoint paths problem. *New Generation Computing* **39**(1), 3–17 (2021)
53. Ruangwises, S., Itoh, T.: Physical zero-knowledge proof for ripple effect. In: Proc. of 15th International Workshop on Algorithms and Computation (WALCOM 2021), LNCS Vol. 12635. pp. 296–307. Springer International Publishing, Cham (2021)
54. Sasaki, T., Miyahara, D., Mizuki, T., Sone, H.: Efficient card-based zero-knowledge proof for sudoku. *Theoretical Computer Science* **839**, 135–142 (2020)
55. Shinagawa, K., Mizuki, T.: The six-card trick: secure computation of three-input equality. In: Proc. of 21st International Conference on Information Security and Cryptology (ICISC 2018), LNCS Vol. 11396. pp. 123–131 (2018)
56. Shinagawa, K., Mizuki, T.: Secure computation of any boolean function based on any deck of cards. In: Proc. of 13th International Workshop on Frontiers in Algorithmics (FAW 2019), LNCS Vol. 11458. pp. 63–75. Springer (2019)
57. Shinagawa, K., Nuida, K.: A single shuffle is enough for secure card-based computation of any boolean circuit. *Discrete Applied Mathematics* **289**, 248–261 (2021)
58. Shinoda, Y., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based covert lottery. In: Proc. of 13th International Conference on Information Technology and Communications Security (SecITC 2020), LNCS Vol. 12596. pp. 257–270. Springer (2020)
59. Shirouchi, S., Nakai, T., Iwamoto, M., Ohta, K.: Efficient card-based cryptographic protocols for logic gates utilizing private permutations. In: Proc. of 2017 Symposium on Cryptography and Information Security (SCIS 2017). pp. 1A2–2 (2017), (In Japanese)
60. Takashima, K., Abe, Y., Sasaki, T., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based protocols for secure ranking computations. *Theoretical Computer Science* **845**, 122–135 (2020)
61. Takashima, K., Miyahara, D., Mizuki, T., Sone, H.: Actively revealing card attack on card-based protocols. *Natural Computing* pp. 1–14 (2021)
62. Toyoda, K., Miyahara, D., Mizuki, T., Sone, H.: Six-card finite-runtime xor protocol with only random cut. In: Proc. of the 7th ACM Workshop on ASIA Public-Key Cryptography. pp. 2–8 (2020)
63. Watanabe, Y., Kuroki, Y., Suzuki, S., Koga, Y., Iwamoto, M., Ohta, K.: Card-based majority voting protocols with three inputs using three cards. In: Proc. of 2018 International Symposium on Information Theory and Its Applications (ISITA). pp. 218–222. IEEE (2018)
64. Yasunaga, K.: Practical card-based protocol for three-input majority. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E103.A**(11), 1296–1298 (2020). <https://doi.org/10.1587/transfun.2020EAL2025>