How to Implement a Random Bisection Cut*

Itaru Ueda[1], Akihiro Nishimura[1], Yu-ichi Hayashi[2], Takaaki Mizuki[3], and Hideaki Sone[3]

[1] Graduate School of Information Sciences, Tohoku University
6–3–09 Aramaki-Aza-Aoba, Aoba, Sendai 980–8579, Japan
{itaru.ueda.t6,akihiro.nishimura.p3}[atmark]dc.tohoku.ac.jp
[2] Faculty of Engineering, Tohoku Gakuin University
1–13–1 Chuo, Tagajo, Miyagi 985–8537, Japan
[3] Cyberscience Center, Tohoku University
6–3 Aramaki-Aza-Aoba, Aoba, Sendai 980–8578, Japan
tm-paper+cardsebi[atmark]g-mail.tohoku-university.jp

Abstract. By using a deck of cards, it is possible to realize a secure computation. In particular, since a new shuffling operation, called a random bisection cut, was devised in 2009, many efficient protocols have been designed. The shuffle functions in the following manner. A sequence of cards is bisected, and the two halves are swapped randomly. This results in two possible cases, depending on whether the two halves of the card sequence are swapped or not. Because there are only two possibilities when a random bisection cut is performed, it has been suggested that information regarding the result of the shuffle could sometimes be leaked visually. Thus, in this paper we propose some methods for implementing a random bisection cut without leaking such information.

Keywords: Cryptography, Card-based protocols, Real-life hands-on cryptography, Secure multi-party computations

1 Introduction

It is known that by using a deck of cards, we can realize secure computations. For example, consider a secure AND computation of bits $a \in \{0, 1\}$ and $b \in \{0, 1\}$, i.e., assume that we would only like to know the value of $a \wedge b$. By utilizing a black card $\boxed{\clubsuit}$ and a red card $\boxed{\heartsuit}$, we can represent the value of a bit as follows:

$$\boxed{\clubsuit}\boxed{\heartsuit} = 0, \ \boxed{\heartsuit}\boxed{\clubsuit} = 1.$$

According to this encoding, each of the input bits a and b can be represented by two face-down cards of different colors:

$$\underbrace{\boxed{?}\boxed{?}}_{a}\ \underbrace{\boxed{?}\boxed{?}}_{b}.$$

* This paper appears in Proceedings of TPNC 2016. The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-49001-4_5.

Table 1: Some of committed AND protocols

	# of colors	# of cards	Type of shuffle	Avg. # of trials
Crépeau and Kilian [2]	4	10	RC	6
Niemi and Renvall [12]	2	12	RC	2.5
Stiglic [15]	2	8	RC	2
Mizuki and Sone [10]	2	6	RBC	1

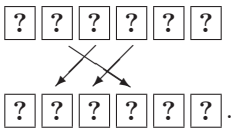
RC: Random Cut, RBC: Random Bisection Cut

A pair of face-down cards (such as in the above example) is called *a commitment*. That is, the two cards on the left constitute a commitment to a, and the two cards on the right constitute a commitment to b. As in this example, the cards we use are either black cards $\boxed{\clubsuit}$ or red cards $\boxed{\heartsuit}$, whose backs are assumed to be identical $\boxed{?}$. As shown in Table 1, many protocols have been designed to perform a secure AND computation, among which we now introduce the Mizuki-Sone AND protocol [10], designed in 2009. Given commitments to inputs a and b along with two additional cards $\boxed{\clubsuit}\boxed{\heartsuit}$, the protocol works as follows.

1. A commitment to 0 is placed between the two input commitments:

$$\underbrace{\boxed{?}\boxed{?}}_{a}\boxed{\clubsuit}\boxed{\heartsuit}\underbrace{\boxed{?}\boxed{?}}_{b} \rightarrow \underbrace{\boxed{?}\boxed{?}}_{a}\underbrace{\boxed{?}\boxed{?}}_{0}\underbrace{\boxed{?}\boxed{?}}_{b}.$$

2. Rearrange the sequence order as follows:

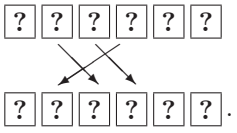


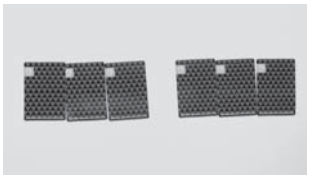
3. Apply *a random bisection cut*:

$$\left[\boxed{?}\boxed{?}\boxed{?}\middle|\boxed{?}\boxed{?}\boxed{?}\right] \rightarrow \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

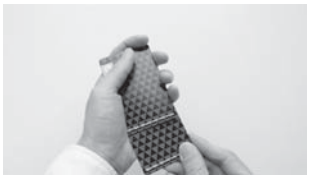
A random bisection cut is a shuffling operation that bisects a sequence of cards and swaps the two halves randomly. Therefore, the shuffle results in two possible cases, depending on whether the two halves are swapped or not, each with a probability of 1/2.

4. Rearrange the sequence order as follows:





(a) Bisect a sequence of cards



(b) Shuffle the two halves

Fig. 1: Execution of a random bisection cut

5. Turn over the two left-most cards. Then, we are able to obtain a commitment to $a \wedge b$ as follows:

$$\clubsuit\,\heartsuit\,\underbrace{?\,?}_{a\wedge b}\,?\,? \quad \text{or} \quad \heartsuit\,\clubsuit\,?\,?\,\underbrace{?\,?}_{a\wedge b}.$$

Although we omit an explanation regarding the correctness and secrecy of this protocol, one can confirm that it outputs a commitment to $a \wedge b$ using six cards after one execution of the random bisection cut [10]. (A protocol such as this that outputs commitments is called a *committed protocol*.)

In practice, humans can perform a random bisection cut by shuffling the two halves after bisecting a given sequence of cards, as illustrated in Figure 1. Thus, given a sequence of six cards

$$\overset{1}{?}\,\overset{2}{?}\,\overset{3}{?}\,\overset{4}{?}\,\overset{5}{?}\,\overset{6}{?},$$

a random bisection cut results in

$$\overset{1}{?}\,\overset{2}{?}\,\overset{3}{?}\,\overset{4}{?}\,\overset{5}{?}\,\overset{6}{?} \quad \text{or} \quad \overset{4}{?}\,\overset{5}{?}\,\overset{6}{?}\,\overset{1}{?}\,\overset{2}{?}\,\overset{3}{?},$$

with a probability of 1/2 for each possibility, where the numbers attached to the cards are for the sake of convenience.

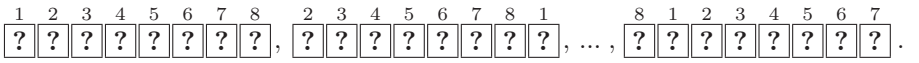
Following the computational model formalized in [4, 8], this random bisection cut can be described as follows:

$$(\mathsf{shuffle}, \{\mathsf{id}, (1\,4)(2\,5)(3\,6)\}),$$

where id represents the identity permutation, and an expression such as $(1\,4)$ represents a cyclic permutation. Therefore, id indicates that the two halves are not swapped, and the permutation $(1\,4)(2\,5)(3\,6)$ indicates that the two halves are swapped. Historically, random bisection cuts first appeared when a six-card AND protocol was designed in 2009 [10]. Even before that, some committed AND protocols had been designed. These earlier protocols employed the *random cut* as a shuffling operation, as shown in Table 1. A random cut refers to a cyclic shuffling operation. For example, given eight face-down cards

$$\overset{1}{?}\,\overset{2}{?}\,\overset{3}{?}\,\overset{4}{?}\,\overset{5}{?}\,\overset{6}{?}\,\overset{7}{?}\,\overset{8}{?},$$

a random cut results in one of the following eight cases, each with a probability of 1/8:



Therefore by following the computational model in [4, 8], we can similarly describe the random cut as

$$(\mathsf{shuffle}, \{\mathsf{id}, \pi, \pi^2, \pi^3, \pi^4, \pi^5, \pi^6, \pi^7\}),$$

where $\pi = (8\,7\,6\,5\,4\,3\,2\,1)$.

Table 2: Some other protocols

	# of colors	# of cards	Type of shuffle	Avg. # of trials
◦ *Non-committed AND Protocols*				
den Boer [1]	2	5	RC	1
Mizuki-Kumamoto-Sone [7]	2	4	RBC	1
◦ *Committed XOR Protocols*				
Crépeau-Kilian [2]	4	14	RC	6
Mizuki-Uchiike-Sone [11]	2	10	RC	2
Mizuki-Sone [10]	2	4	RBC	1

As seen in Table 1, committed AND computations have become more efficient by virtue of the introduction of the random bisection cut in 2009. This introduction also provided the additional benefit that we were able to improve the efficiency of non-committed AND computations and committed XOR computations, as detailed in Table 2. In addition, further efficient protocols have been designed using random bisection cuts [3, 5, 6, 13, 14].

As explained above, *card-based protocols* are intended in practice to be executed by humans, who would like to actually perform secure computations using a real deck of cards. Hence, when we execute a card-based protocol, it is expected that all players gather at the same physical location, and perform operations such as shuffles in public, as in the case of ordinary card games [9].

In order to implement a random cut in such a situation, it is sufficient that each player cuts a sequence of face-down cards in turn until all players are satisfied with the result. Indeed, we note that in practice it is relatively easy to implement a random cut such that nobody is able to determine the result of the shuffle at all. We will discuss this further in Section 4.

Meanwhile, when we execute a random bisection cut in reality, as illustrated in Figure 1, there exists a concern that the result of the shuffle may leak, because there are only two possibilities, i.e., the two halves of the card sequence are swapped or not. Therefore, this paper provides some methods for executing a random bisection cut securely.

This paper is composed as follows. In Section 2, we present some methods for implementing a random bisection cut using auxiliary tools. In Section 3, we propose methods to reduce the execution of a random bisection cut to the execution of random cuts using dummy cards. In Section 4, we discuss implementations of the random cut, and confirm through a basic experiment that humans are able to implement random cuts securely, implying that random bisection cuts can also be implemented securely.

2 Executing a Random Bisection Cut Using Auxiliary Tools

In this section, we provide methods for implementing a random bisection cut by using auxiliary tools that consist of everyday objects.

2.1 The Use of Paper Clips, Envelopes, or Boxes

When players operate a random bisection cut, if they are not familiar with playing cards and have difficulty shuffling the two halves such that each half stays together, as in Figure 1(b), then they may want to secure each half using paper clips or envelopes [7, 10]. By using these auxiliary tools, we are able to fix each of the two halves together as shown in Figure 2. Following this, it suffices to swap the two bundles of cards randomly.

However, as explained in Section 1, the result of the shuffle could be revealed when we execute a random bisection cut in public. That is, someone may count how many times the two bundles are swapped. To avoid such a leak of information, one solution is that each player shuffles the two bundles behind his/her back or under a table, so that other players cannot see whether the two bundles are swapped or not. In this case, it may be preferable to use envelopes or boxes (as illustrated in Figure 3) rather than paper clips, to avoid malicious actions.

However, as mentioned in Section 1, it is desirable for all actions to be performed in front of all players and/or third parties publicly. Therefore, in Sections 2.2 and 3 we present implementations of random bisection cuts where every action can be performed completely in public.

Fig. 2: Each half is placed in an envelope

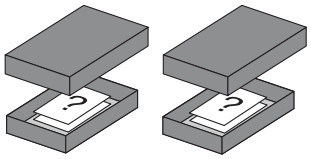


Fig. 3: Each half is placed in a box

(a) Separator and two halves

(b) One half is placed on the separator

(c) The pile consisting of one half and the separator is flipped

(d) The other half is placed on the pile

Fig. 4: Setup for spinning throw

(a) Hold the pile of cards

(b) Throw the pile like a coin

Fig. 5: A spinning throw

2.2 The Use of a Separator Card and Rubber Band

In this subsection, we present a novel method of performing a random bisection cut using a separator card with a rubber band. Both sides of the separator (as shown in the middle of Figure 4 (a)) must be indistinguishable.

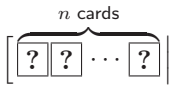
The method works as follows. First, a sequence of cards is bisected, with one half placed on the separator, as shown in Figure 4(b). Second, the pile consisting of one half and the separator is turned over, as in Figure 4(c)[1]. Third, the other half is placed on the pile, as shown in Figure 4(d), and these are fixed together using a rubber band, to prevent the cards from scattering. Next, the pile is thrown in a spinning manner (as illustrated in Figure 5). We call this action a *spinning throw*. After the pile is caught, we are completely unsure of which half is on the top. Finally, the rubber band is removed, and the actions described in Figure 4 are undone in reverse order, from (d) to (a). In this manner, we can conduct a random bisection cut securely.

3 Executing a Random Bisection Cut Using Dummy Cards

In this section, we propose methods for reducing the execution of a random bisection cut to the execution of random cuts using dummy cards.

[1] The separator prevents information regarding the color of cards from being leaked.

It is assumed throughout this section that we want to apply a random bisection cut to a sequence of $2n$ cards, where $n \geq 2$:

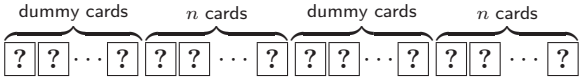


3.1 The Use of Cards of Other Colors

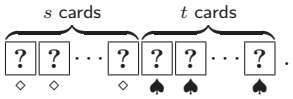
Here, as dummy cards we use cards whose backs are $\boxed{?}$ and faces are different from $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$, namely $\boxed{\diamond}$ or $\boxed{\spadesuit}$. Specifically, we use $2s$ $\boxed{\diamond}$ and $2t$ $\boxed{\spadesuit}$ cards, where $s, t \geq 1$. That is, we have a total of $2(s+t)$ additional cards.

By using such dummy cards, we are able to implement a random bisection cut as follows.

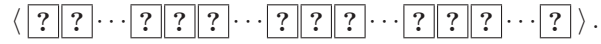
1. Place dummy cards with their faces down, as follows:



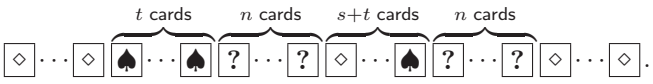
   where the dummy cards are arranged as below:



2. Apply a random cut:



3. Turn over the left-most card.
   (a) If the face-up card is $\boxed{\diamond}$, then turn over cards forward (in the right-hand direction) until t $\boxed{\spadesuit}$ cards appear. Now, we have determined the positions of all of the dummy cards, and hence we can remove them all:



   (b) If the face-up card is $\boxed{\spadesuit}$, then turn over cards backward (aside from cyclic rotations) until s $\boxed{\diamond}$ cards appear. Now, we have determined the positions of all of the dummy cards, and hence we can remove them all:

   (c) If the face-up card is $\boxed{\clubsuit}$ or $\boxed{\heartsuit}$, then turn it over again and return to Step 2.

In this manner, after all of the dummy cards are removed, a random bisection cut has been completed.

In Step 3, the probability that either (a) or (b) occurs is $(s+t)/(n+s+t)$. Therefore, we are able to implement a random bisection cut using $2(s+t)$ dummy cards after an average of $(n+s+t)/(s+t)$ executions of the random cut.

This method of discarding dummy cards was first devised by Crépeau and Kilian [2], when they proposed some random permutation generating protocols. Here, we have adopted their idea.

Regarding the parameters s and t, there is a trade-off between the number of required cards and the average number of executions of the random cut. For example, if we want to implement the Mizuki-Sone six-card AND protocol [10] with an average number of two random cuts, then we require six additional dummy cards, and hence this requires more cards than Stiglic's eight-card AND protocol [15] (although the former might have the advantage that it is simpler to understand its correctness).

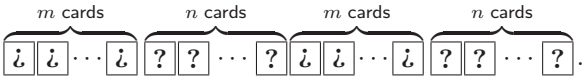
Moreover, we can select the parameters as $s = 1$ and $t = 0$, i.e., the above method works even with only two $\boxed{\diamond}$ dummy cards (and no $\boxed{\spadesuit}$ cards).

3.2 Utilizing Vertical Asymmetricity of the Backs of Cards

In Section 3.1, we required additional types of cards to reduce the execution of a random bisection cut to the execution of random cuts. On the other hand, in this section we do not use such additional cards, but rather employ the same cards that we have used before ($\boxed{\clubsuit}$ and $\boxed{\heartsuit}$) as dummy cards.

Our method works as follows. We exploit the vertical asymmetricity of the backs of cards $\boxed{?}$. Because the back is asymmetric, it can be seen as either $\boxed{?}$ or $\boxed{¿}$, depending on the setting. Now, we describe how to implement a random bisection cut by executing a random cut with $2m$ additional dummy cards, where $m \geq 1$. (Here, any cards of type $\boxed{\heartsuit}$ or $\boxed{\clubsuit}$ can be used as dummy cards.)

1. Arrange the $2m$ additional dummy cards with their upsides facing down, as follows:



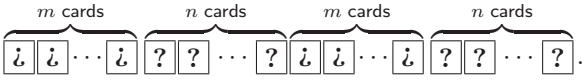
2. Apply a random cut:

$$\langle \boxed{?}\boxed{?}\cdots\boxed{?}\boxed{¿}\boxed{¿}\cdots\boxed{¿}\boxed{?}\boxed{?}\cdots\boxed{?}\boxed{¿}\boxed{¿}\cdots\boxed{¿} \rangle .$$

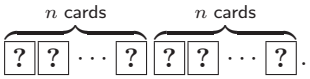
3. Cyclically shift the first several cards to the rightmost positions, without changing their order, so that the first card will be a dummy card:

$$\boxed{?}\cdots\boxed{?}\boxed{¿}\cdots\boxed{¿}\boxed{?}\cdots\boxed{?}\boxed{¿}\cdots\boxed{¿}\boxed{?}\cdots\boxed{?}$$

$$\downarrow$$



4. Remove all dummy cards:



In this manner, by using $2m$ additional cards and executing one random cut, we are able to implement a random bisection cut. For example, the Mizuki-Sone six-card AND protocol [10] can be implemented with one random cut by adding two additional cards.

In this method, we must apply a random cut to cards that have asymmetric backs, and hence information regarding the result of the shuffle could be leaked more easily than with cards that have identical backs.

Taking this into account, we will discuss secure implementations of the random cut in the next section.

4 Secrecy of Implementations of the Random Cut

In Section 3, we proposed some methods for reducing the execution of a random bisection cut to the execution of random cuts. In general, it is believed that a random cut can be securely implemented by humans. To support this belief, we discuss the secure implementation of a random cut by performing a shuffle with a real deck of cards. Specifically, in Section 4.1 we point out that a naive implementation is not secure, and then propose a secure implementation that we call the "Hindu cut." In Section 4.2, we demonstrate that the "Hindu cut" is in fact a secure implementation of a random cut by conducting a basic experiment.

4.1 Discussion Regarding Implementations

Because a random cut consists of a cyclic shuffle, its simple implementation proceeds as in Figure 6: some cards (or a card) are taken from the top of the pile, and then moved to the bottom of the pile (this is called a *cut*). At every cut, we should change the number of cards that are to be moved. To verify whether this simple implementation is secure, we conducted an experiment.

We asked eleven students in our laboratory to observe one author executing a simple implementation of a random cut with a pile of eight cards. We then asked them to track a specific card.

While most of the participants were unable to recognize the location of the targeted card and gave up guessing, three participants were able to follow the move and track the specific card with a high probability. Therefore, in the presence of people who are capable of following the cutting move, such a simple implementation is not secure.

Thus, we require an alternative secure implementation of the random cut. The three participants who were able to break the simple implementation informed us that they could visually observe how many cards were moved at every cut, and summed up the numbers. Hence, the key consideration is how to make it impossible for people to count the number of cards moved during every cut.

Fig. 6: Simple execution of a random cut

Fig. 7: Execution of a "Hindu cut"

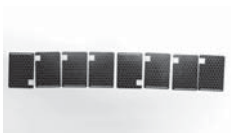


Fig. 8: A card sequence including two upside-down cards

One idea is to move cards from the bottom to the top of the pile instead of moving them from the top to the bottom when executing a cut operation. By doing so, it becomes difficult to recognize the number of cards that have been moved. Moreover, if the positions of the cards are out of alignment, as in Figure 6, then it is possible to easily recognize the number of cards moved. Therefore, we should ensure that cards are not out of alignment when we execute cut operations.

Based on this idea, we found that a variation of the so-called Hindu shuffle (shown in Figure 7) is effective for preventing the cut operation from being revealed (we call this the *Hindu cut*). In fact, the three participants mentioned above were unable to guess the specific cards, and hence they all gave up following the implementation of a Hindu cut.

Next, in order to verify the security of the Hindu cut, we conducted another experiment based on the method presented in Section 3.2, which exploits the vertical asymmetricity of the backs of cards. That is, we placed two cards (out of eight) with their upsides down and applied a Hindu cut, as illustrated in Figure 8. Clearly, these upside-down cards are advantageous to people who wish to track the move. Furthermore, we asked the participants to guess the location of a specific upside-down dummy card out of the two. (There are only two possibilities after the shuffle, by virtue of the two upside-down cards.) As a result, even the three participants mentioned above gave up guessing the specific dummy card.

4.2 Confirming the Security of the Hindu Cut

In order to confirm the security of the Hindu cut more, we requested 72 participants (as shown in Table 3) who came to the Sone-Mizuki laboratory booth in the Open Campus of Tohoku University in 2016, to watch a movie depicting the execution of a Hindu cut. In the movie, we employ the same sequence of cards as in Section 4.1. We arranged two upside-down cards, named A and B, and applied

Table 3: Number of examinees

# of participants	ratio of Male and Female (M:F)
72	62 : 10

Table 4: Result of our experiment

Choice	# of answers
(1) I have no idea.	64
(2) Definitely, it must be A.	5
(3) Definitely, it must be B.	3

a Hindu cut. The time taken for the execution was 30 seconds, and the movie can be found at `https://youtu.be/Zm-IpuOobIY`. Following the Hindu cut, we asked every participant whether he/she could determine which card was the left upside-down card with certainty. Each participant was asked to choose his/her answer from the following three options.

(1) I have no idea[2].
(2) Definitely, it must be A.
(3) Definitely, it must be B.

The result is presented in Table 4. Most of the participants submitted the answer (1), whereas five people gave the answer (2), and three gave the answer (3)[3]. For the participants whose answers were (2) or (3), we requested that they watch additional movies and answer again, in order to rule out wild guesses. As a result, none of the participants were able to answer correctly for all of the five movies that we had prepared.

Thus, we can conclude that the Hindu cut is an effective method for implementing a random cut securely, although we recognize that a more careful and wide ranging investigation is still required.

5 Conclusion

The random bisection cut has played an important role in improving card-based protocols. However, implementation issues have not previously been discussed. Therefore, in this paper we have proposed some novel methods for implementing the random bisection cut, and demonstrated that humans are able to implement it in practice.

Acknowledgments

We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper. We would like to offer our special thanks to Kohei Yamaguchi, who provided an excellent implementation of the random bisection cut, the spinning throw, as introduced in Section 2.2. In addition, we

[2] If a participant could not track the move with confidence, then he/she is assumed not to have any motivation to reveal secret information from the result of the shuffle.
[3] We note that the correct answer was B.

are grateful to all members of the Sone-Mizuki laboratory in Tohoku University, who cooperated with our experiment in Section 4. This work was supported by JSPS KAKENHI Grant Number 26330001.

References

1. den Boer, B.: More efficient match-making and satisfiability: the five card trick. In: Quisquater, J.J., Vandewalle, J. (eds.) Advances in Cryptology — EUROCRYPT '89, Lecture Notes in Computer Science, vol. 434, pp. 208–217. Springer Berlin Heidelberg (1990)
2. Crépeau, C., Kilian, J.: Discreet solitary games. In: Stinson, D.R. (ed.) Advances in Cryptology — CRYPTO '93, Lecture Notes in Computer Science, vol. 773, pp. 319–330. Springer Berlin Heidelberg (1994)
3. Ishikawa, R., Chida, E., Mizuki, T.: Efficient card-based protocols for generating a hidden random permutation without fixed points. In: Calude, C.S., Dinneen, M.J. (eds.) Unconventional Computation and Natural Computation, Lecture Notes in Computer Science, vol. 9252, pp. 215–226. Springer International Publishing (2015)
4. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Iwata, T., Cheon, J. (eds.) Advances in Cryptology – ASIACRYPT 2015, Lecture Notes in Computer Science, vol. 9452, pp. 783–807. Springer Berlin Heidelberg (2015)
5. Mizuki, T.: Card-based protocols for securely computing the conjunction of multiple variables. Theoretical Computer Science 622, 34–44 (2016)
6. 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.) Unconventional Computation and Natural Computation, Lecture Notes in Computer Science, vol. 7956, pp. 162–173. Springer Berlin Heidelberg (2013)
7. Mizuki, T., Kumamoto, M., Sone, H.: The five-card trick can be done with four cards. In: Wang, X., Sako, K. (eds.) Advances in Cryptology — ASIACRYPT 2012, Lecture Notes in Computer Science, vol. 7658, pp. 598–606. Springer Berlin Heidelberg (2012)
8. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. International Journal of Information Security 13(1), 15–23 (2014)
9. Mizuki, T., Shizuya, H.: Practical card-based cryptography. In: Ferro, A., Luccio, F., Widmayer, P. (eds.) Fun with Algorithms, Lecture Notes in Computer Science, vol. 8496, pp. 313–324. Springer International Publishing (2014)
10. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) Frontiers in Algorithmics, Lecture Notes in Computer Science, vol. 5598, pp. 358–369. Springer Berlin Heidelberg (2009)
11. Mizuki, T., Uchiike, F., Sone, H.: Securely computing XOR with 10 cards. The Australasian Journal of Combinatorics 36, 279–293 (2006)
12. Niemi, V., Renvall, A.: Secure multiparty computations without computers. Theoretical Computer Science 191(1–2), 173–183 (1998)
13. Nishida, T., Hayashi, Y., Mizuki, T., Hideaki, S.: Securely computing three-input functions with eight cards. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences 98(6), 1145–1152 (2015)
14. Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols for any boolean function. In: Jain, R., Jain, S., Stephan, F. (eds.) Theory and Applications of Models of Computation, Lecture Notes in Computer Science, vol. 9076, pp. 110–121. Springer International Publishing (2015)

15. Stiglic, A.: Computations with a deck of cards. Theoretical Computer Science 259(1–2), 671–678 (2001)