# Optimally-Fair Exchange of Secrets via Delay Encryption and Commutative Blinding

Ivo Maffei(✉) and Andrew W. Roscoe

Department of Computer Science, University of Oxford, Oxford, UK
{ivo.maffei,bill.roscoe}@cs.ox.ac.uk

**Abstract.** We propose a new fair exchange protocol that takes advantage of delay encryption and commutative encryption to achieve optimal partial fairness among all protocols involving one-way messages. Our protocol consists of 3 setup messages and $2N + 1$ exchange messages and it is fair against covert adversaries with probability $1 - \frac{1}{2N}$. We prove that this is optimal up to shortening the setup phase which is notably more efficient than existing protocols.

**Keywords:** Fair Exchange · Timed-Release Encryption · Partial Fairness · Commutative Blinding

## 1 Introduction

A fair exchange protocol allows two parties to exchange secrets fairly without mutual trust. Given its practical importance, this problem has been studied extensively in different contexts. Many solutions using Trusted Third Parties as mediators (e.g. [10]) or judges (e.g. [1]) have been proposed. When in lack of a TTP, fair exchange was first shown to be impossible by Even and Yacobi [9] in 1980 and later by Pagnia and Gärtner [21]. Garbinato and Rickebusch [11] analysed this result further to provide a finer classification when some trust is present in the network. While their result is very interesting in the setting where multiple parties are involved, in our 2-party context they conclude that one party must be trusted.

Similar problems such as multi-party coin flipping or exchange of signatures were also proved impossible, respectively, by Cleve [5] and by Even and Yacobi [9]. As a result, researchers tried to design protocols that either achieve weaker fairness properties or take advantage of stronger assumptions. For instance, under the assumption that both parties have similar computational power, bit commitments schemes [7] can be used to achieve weaker fairness where a party's

This research was funded in whole, or in part, by UKRI 2421791 and Crypto.com. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

F. Baldimtsi and C. Cachin (Eds.): FC 2023, LNCS 13950, pp. 94–111, 2024.
https://doi.org/10.1007/978-3-031-47754-6_6

knowledge is at most one bit more than the other party's knowledge. Using the weaker assumption that both parties have similar sequential computational power, Boneh and Naor [3] proposed a protocol where bit commitments were replaced with timed commitments. Alternatively, Ben-Or *et al.*[2] proposed a fair exchange of signatures where the probability that one party is committed to a contract while the other is not can be arbitrarily small. Following the same trend of using probabilistic definitions of fairness, Gordon and Katz [16] introduced the concept of partial fairness for secure two-party computations.

A very similar concept was used in the context of two-party fair exchange by Roscoe and Ryan [24] and later refined by Couteau *et al.* [6]. We follow their approach and propose a fair exchange protocol where the probability of an unfair run is almost inversely proportional to the number of messages exchanged. This fairness is achieved against covert adversaries, i.e. the malicious entity will not perform actions for which they can be blamed. We achieve a stronger form of fairness and better bounds than the protocols of [6,16]. In particular, Gordon and Katz [16] show that general 2 party computations cannot be performed with polynomial partial fairness when parties are allowed to early abort the protocol and only polynomial running time. Couteau *et al.* [6] escape this impossibility result using timed-release encryption and show that any 2 party functionality can achieve partial fairness by designing a partially fair exchange protocol.

### 1.1 Our Contribution and Related Work

The key idea used in the protocol by Couteau *et al.* as well as in [16] is to hide the secret among a set of dummy values, then exchange them one by one. In this paper, we explore the theoretical limits of this approach by proving an upper bound on the partial fairness that can be achieved and then building a protocol that reaches it. In this paper, the results and protocols of that approach are improved significantly.

In the protocol analysed by Couteau *et al.*, the probability that Alice receives Bob's secret is at most $\frac{2}{N}$ greater than Bob's chances of receiving Alice's secret (and vice versa), where $N$ is the number of messages in the exchange phase of their protocol. However, each party could get roughly a $\frac{1}{4}$ probability of receiving the other party's secret without revealing theirs. In our protocol, the latter probability is bounded above by $\frac{1}{N-1}$.

Similarly, Gordon and Katz's [16] protocol hides the correct output of the computation among a list of other random values. Our protocol improves on theirs in three distinct ways. Firstly, in [16] each party has immediate access to all the value already exchanged. Therefore, if a malicious entity had access to some information on the value of the correct output, they could recognise some random values as such. Hence, their decision on when to abort will no longer be uniformly random. We solve this by encrypting the secrets to exchange and only work with encryption keys. Assuming that the honest party picks truly random keys, the adversary cannot distinguish which key is used for the secrets until the ciphertext is available at the end of the protocol. Secondly, we propose a concrete setup phase consisting of only 3 messages, of which the last will be sent together with the first exchange message. In [16], the sub-protocol preceding the exchange phase is only described abstractly. Finally, we achieve better fairness, i.e. we halve the probability of an unfair run of the protocol.

Like those of [6,24], our protocol relies on the unusual assumptions of the existence of delay encryption (often called Timed-Release Encryption) [23] and a commutative blinding scheme. The former assumption is what let our protocol circumvent the impossibility result proved by Gordon and Katz [16]. Only a few delay encryption schemes have been proposed in the past 30 years. We believe exponentiation modulo a prime [18] is a viable option in our case since it relies only on the assumption that repeated squaring is inherently sequential and that both parties have a somewhat similar sequential computation power. This is much more reasonable assumptions than similarity in general (i.e. parallel) computational power.

In Sect. 2, we present the preliminary definitions of each cryptographic primitive used in the fair exchange protocol. The next section is dedicated to the description of our protocol. In Sect. 4, we analyse the fairness of the proposed protocol as well as showing its optimality.

## 2 Preliminaries

In this section, we lay out the definitions of the cryptographic primitives that will be used in the fair exchange protocol.

### 2.1 Notation

In this paper we will use the following notation:

- $\mathbb{Z}_N$ is the *set* $\{0, \dots, N-1\}$
- vectors are written in bold. $\boldsymbol{v}[i]$ is the $i^{\text{th}}$ entry of $\boldsymbol{v}$ starting from 0.
- $(a_i)_{i \in X}$ is a vector of length $|X|$ whose entries are $a_i$
- If $\boldsymbol{v} \in X^N$ and $f : X \to Y$, we write $f(\boldsymbol{v})$ for $(f(\boldsymbol{v}[i]))_{i \in \mathbb{Z}_N} \in Y^N$.
- $x \overset{\$}{\leftarrow} X$ means that $x$ is sampled uniformly from the set $X$.
- mod is the usual binary function $\mathbb{Z} \times \mathbb{N}^+ \to \mathbb{N}$ such that $a \bmod b \in \mathbb{Z}_b$.
- $a \| b$ is the concatenation of $a$ and $b$ by interpreting them as binary strings.

### 2.2 Symmetric Encryption and Hashing

We use standard symmetric encryption $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ [15,17] and only require it to be secure against ciphertext-only attacks. We write $\mathsf{Enc}_k(m)/\mathsf{Dec}_k(m)$ to mean the encryption/decryption of message/ciphertext $m$ under the key $k$.

Our protocol takes advantage of cryptographic hashing $(\mathsf{H} : \{0,1\}^* \to \{0,1\}^n)$ [15,17] as a commitment scheme. Therefore, both preimage resistant and second preimage resistant are required.

### 2.3 Delay Encryption

Delay encryption (sometimes called Timed-Release Encryption) is an unusual cryptographic primitive whose aim is not to provide confidentiality of a message from other parties, but to hide the message from anyone for some predefined amount of time. For the reader accustomed with Timed-Release Encryption, what we use and define is a "delay" time-lock-puzzle-based encryption scheme rather than time-specific schemes using trusted third parties. This is justified because, in the presence of a TTP, the fair exchange problem becomes trivial.

More formally, a delay encryption scheme is a triple of algorithms and associated sets $\mathcal{M}, \mathcal{C}, \mathcal{P}$:

$$\mathsf{Pgen}\colon \{1\}^* \times \mathbb{N} \to \mathcal{P} \qquad \mathsf{delay}\colon \mathcal{M} \times \mathcal{P} \to \mathcal{C} \qquad \mathsf{open}\colon \mathcal{C} \times \mathcal{P} \to \mathcal{M}$$

Intuitively, $\mathsf{Pgen}(1^\lambda, T)$ generates public parameters $\mathfrak{p}$ that are used to delay a message for time $T$. $\mathsf{delay}_T(m)$ will mean that the algorithm is used on $m$ to delay it for elapsed time $T$. Similarly, we omit $\mathfrak{p}$ and write $\mathsf{open}(c)$. Practically speaking, $\mathsf{delay}_T(m)$ will create a puzzle $c$ so that $\mathsf{open}(c)$ can solve the puzzle and obtain $m$ only after at least $T$ (sequential) time. A honest party with moderate computational power should be able to run $\mathsf{open}(c)$ in sequential time not much longer than $T$. In order for our scheme to make sense, we set the following requirement

$$\forall m \in M \; \forall T \in \mathbb{N} \quad \mathsf{open}(\mathsf{delay}_T(m)) = m$$

We say that a delay encryption is COA-secure if for any family of circuits $\mathcal{A}$ of conceivable size and depth at most $\lambda T$, we have

$$\Pr_{\substack{m \xleftarrow{\$} \mathcal{M} \\ T \xleftarrow{\$} \mathbb{N}}} \left[ m \leftarrow \mathcal{A}(c, T, \mathfrak{p}) \;\middle|\; c \leftarrow \mathsf{delay}_T(m) \wedge \mathfrak{p} \leftarrow \mathsf{Pgen}(1^\lambda, T) \right] < \frac{1}{|\mathcal{M}|} + \mathsf{negl}\,(\lambda)$$

Intuitively, a COA-secure delay encryption scheme correctly hides encrypted messages for the expected amount of time. We remark that the size of such circuits will depend on the current state of technology. As noted in [18], allowing all polynomially-sized circuits could lead to misleading results with circuits much larger than what is feasible at the time of writing.

### 2.4 Commutative Blinding

A feature of our protocol is the use of commutative blinding that enables two parties to jointly shuffle a deck of cards in such a way that neither know where a given card is. Usually a blinding scheme is nothing but an encryption scheme, however the usual definition of a commutative encryption scheme is stricter than what we need. As a result, we define a commutative blinding scheme as a tuple of algorithms $(\mathsf{KGen}_1, \mathsf{KGen}_2, \mathsf{Blind}_1, \mathsf{Blind}_2, \mathsf{Unblind}_1, \mathsf{Unblind}_2)$ with sets $(\mathcal{M}, \mathcal{K}_1, \mathcal{K}_2, \mathcal{C}_{\text{int}}, \mathcal{C})$ such that

$$\begin{array}{lll} \mathsf{KGen}_1\colon \{1\}^* \to \mathcal{K}_1 & \mathsf{Blind}_1\colon \mathcal{M} \times \mathcal{K}_1 \to \mathcal{C}_{\text{int}} & \mathsf{Unblind}_1\colon \mathcal{C} \times \mathcal{K}_1 \to \mathcal{C}_{\text{int}} \\ \mathsf{KGen}_2\colon \{1\}^* \to \mathcal{K}_2 & \mathsf{Blind}_2\colon \mathcal{C}_{\text{int}} \times \mathcal{K}_2 \to \mathcal{C} & \mathsf{Unblind}_2\colon \mathcal{C}_{\text{int}} \times \mathcal{K}_2 \to \mathcal{M} \end{array}$$

In order for the scheme to make sense we require

$$\forall k_1 \in \mathcal{K}_1 \; \forall k_2 \in \mathcal{K}_2 \; \forall m \in \mathcal{M} \qquad \mathsf{Unblind}_2\left(\mathsf{Unblind}_1\left(\mathsf{Blind}_2\left(\mathsf{Blind}_1\left(m, k_1\right), k_2\right), k_1\right), k_2\right) = m$$

If these functions were used in other contexts, these names could be misleading. In particular, $\mathsf{Unblind}_1$ and $\mathsf{Blind}_1$ are not necessarily inverses, nor are $\mathsf{Unblind}_2$ and $\mathsf{Blind}_2$. In most practical scenarios, the blinding and unblinding will be inverses as well as $\mathsf{Blind}_1 = \mathsf{Blind}_2$ and $\mathsf{Unblind}_1 = \mathsf{Unblind}_2$. However, this is not required and constructions using homomorphic encryption are likely to need the full generality of our definition.

We say that $\mathsf{Blind}_1$ is COA-secure if for any PPT adversary $\mathcal{A}$ we have

$$\Pr_{\substack{m \xleftarrow{\$} \mathcal{M} \\ k \xleftarrow{\$} \mathsf{KGen}_1(1^\lambda)}} \left[ m \leftarrow \mathcal{A}(c) \;\middle|\; c \leftarrow \mathsf{Blind}_1(m, k) \right] < \frac{1}{|\mathcal{M}|} + \mathsf{negl}\left(\lambda\right)$$

The remaining security requirements are described in Figs. 1, 2 and 3.

We say that an adversary wins the $N$-KPA game (Fig. 1) if $m' = m_{N+1}$. In the $(N, \mathcal{P})$-CPA game, we note that $\mathcal{P}$ is a set of permutations of $N$-dimensional vectors. We say that an adversary wins the $(N, \mathcal{P})$-CPA game if $\sigma' = \sigma$. In other words, if we are presented a blinded permutation of known distinct messages, we should not be able to tell which message corresponds to which. Finally, we say that an adversary wins the KIND game if $b' = b$.

We say that $\mathsf{Blind}_1$ is $N$-KPA secure if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the $N$-KPA game is at most $\frac{1}{|\mathcal{M}|-N} + \mathsf{negl}\left(\lambda\right)$.

We say that $\mathsf{Blind}_2$ is $(N, \mathcal{P})$-CPA secure if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the $(N, \mathcal{P})$-CPA game is at most $\frac{1}{|\mathcal{P}|} + \mathsf{negl}\left(\lambda\right)$.

We say that $\mathsf{Blind}_2$ is KIND secure if for any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the KIND game is at most $\frac{1}{2} + \mathsf{negl}\left(\lambda\right)$.

Explanations for these requirements will be discussed in Sect. 3.2 after we present our protocol.

### 2.5 Fair Exchange

We say that a protocol $\Pi$ is an ideal fair exchange of secrets between two parties Alice and Bob if it achieves the same result as performing the exchange in an ideal world where there is a TTP collecting the secrets and exchanging them. Intuitively, the exchange is *fair* if Alice learns Bob's s secret if and only if Bob learns Alice's secret. As noted in the introduction, an ideal fair exchange protocol is not possible without a TTP or some very strong assumptions on the network of parties. As a result, our fair exchange protocol achieves fairness only in a probabilistic way.

**Definition 1.** *We say that a probabilistic protocol $\Pi$ achieves $p$-partial fairness if the probability of an unfair run is at most $p$.*

$N$-KPA security for $\mathsf{Blind}_1$

| **Adversary** | | **Oracle** |
|---|---|---|
| | | $\boldsymbol{m} \xleftarrow{\$} \mathcal{M}^{N+1}; k \xleftarrow{\$} \mathcal{K}$ |
| | | $\boldsymbol{c} \leftarrow \mathsf{Blind}_1(\boldsymbol{m}, k)$ |
| | $\xleftarrow{\boldsymbol{c}[N+1], (\boldsymbol{m}[i], \boldsymbol{c}[i])_{i \in \mathbb{Z}_N}}$ | |
| **return** $m'$ | | |

**Fig. 1.** $N$-KPA game

$(N, \mathcal{P})$-CPA security for $\mathsf{Blind}_1$ and $\mathsf{Blind}_2$

| **Adversary** | | **Oracle** |
|---|---|---|
| $\boldsymbol{m} \leftarrow \mathcal{M}^N; k_{\mathcal{A}} \leftarrow \mathcal{K}$ | | |
| $\boldsymbol{c} = \mathsf{Blind}_1(\boldsymbol{m}, k_{\mathcal{A}})$ | | |
| | $\xrightarrow{\boldsymbol{c}}$ | |
| | | $\boldsymbol{k} \xleftarrow{\$} \mathcal{K}^N; \sigma \xleftarrow{\$} \mathcal{P}$ |
| | | $(d_i)_{i \in \mathbb{Z}_N} = \sigma\left((\mathsf{Blind}_2(\boldsymbol{c}[i], \boldsymbol{k}[i]))_{i \in \mathbb{Z}_N}\right)$ |
| | $\xleftarrow{\boldsymbol{d}}$ | |
| **return** $\sigma'$ | | |

**Fig. 2.** $(N, \mathcal{P})$-CPA game

KIND security for $\mathsf{Blind}_2$

| **Adversary** | | **Oracle** |
|---|---|---|
| | | $\boldsymbol{m} \xleftarrow{\$} \mathcal{M}^2; k_O \xleftarrow{\$} \mathcal{K}$ |
| | | $\boldsymbol{c} \leftarrow \mathsf{Blind}_1(\boldsymbol{m}, k_O)$ |
| | $\xleftarrow{\boldsymbol{c}}$ | |
| $\boldsymbol{k} \xleftarrow{\$} \mathcal{K}^2; \boldsymbol{d} \leftarrow (\mathsf{Blind}_2(\boldsymbol{c}[i], \boldsymbol{k}[i]))_{i \in \{0,1\}}$ | | |
| | $\xrightarrow{\boldsymbol{d}}$ | |
| | | $\boldsymbol{e} \leftarrow \mathsf{Unblind}_1(\boldsymbol{d}, k_O); b \xleftarrow{\$} \{0,1\}$ |
| | $\xleftarrow{e_b}$ | |
| **return** $b'$ | | |

**Fig. 3.** KIND game

In other words, a malicious entity can obtain the other party's secret without revealing theirs with probability at most $p$. We also say that $\Pi$ achieves fairness with probability $p$ to mean that it is $(1-p)$-partially fair.

# 3 Protocol

Firstly, we define a few permutations that will be used in the protocol.

$$\begin{aligned} \sigma_d &: \mathbb{Z}_N \to \mathbb{Z}_N \\ x &\mapsto x + d \mod N \end{aligned} \qquad \begin{aligned} \sigma_d^e &: \mathbb{Z}_{N+1} \to \mathbb{Z}_{N+1} \\ x &\mapsto \begin{cases} \sigma_d(x) & \text{if } 0 \leq x < N \\ N & \text{if } x = N \end{cases} \end{aligned}$$

$$\begin{aligned} \tau_N^b &: X^N \to X^N \\ \boldsymbol{v} &\mapsto \left(\boldsymbol{v}[i(-1)^b + (N-1)b]\right)_{i \in \mathbb{Z}_N} \end{aligned} \qquad \begin{aligned} \pi^b &: X^{N+1} \to X^{N+1} \\ \boldsymbol{v} &\mapsto (\boldsymbol{v}[(i-b) \bmod (N+1)])_{i \in \mathbb{Z}_{N+1}} \end{aligned}$$

Figure 4 is the exchange protocol.

## 3.1 Protocol Overview

Our protocol follows the idea used by Roscoe and Ryan [24] and Couteau *et al.*[6], i.e. Alice and Bob will hide their secret among a set of dummy messages. The aim is to prevent any party from predicting when the secrets will be exchanged or distinguish when they have sent or received a secret until all exchanges should have finished. A key component to achieve this is delay encryption. The delayed messages $D_A, D_B$ fulfil two different purposes. Firstly, they guarantee the exchanged secrets cannot be computed as soon as the keys $\boldsymbol{k^A}[0], \boldsymbol{k^B}[0]$ are received, but only at the end of the protocol. Secondly, they behave as a timed commitment to each player's strategy. This is used to detect any active cheater[1]. Most of the complexity of the protocol lies in the three setup messages that are used to hide $\boldsymbol{k^A}[0]$ and $\boldsymbol{k^B}[0]$ among dummy keys. The shuffling process proceeds as follow:

1. Each party places their secret among a list of $N$ values and permutes it. Alice places an extra dummy value at the end of her list.
2. After exchanging these lists, each party permutes the received list.
3. Bob decides whether to place Alice's extra dummy at the start or end of Alice's list.
4. Alice decides whether to reflect each list around their midpoint or not.

[1] For example, if its initially declared and committed strategy obliges A to send its own secret in message $x$, then not doing so would be a detectable abuse even if the protocol is terminated before its end.

Optimally Fair Exchange Protocol

**Alice** **Bob**

Setup phase

$d_A \xleftarrow{\$} \mathbb{Z}_N; \boldsymbol{k^A} \xleftarrow{\$} \mathcal{K}^{N+1}$

$\boldsymbol{B^A} \xleftarrow{\$} \mathcal{BK}^N; B_c^A \xleftarrow{\$} \mathcal{BK}$

$\boldsymbol{c^A} \leftarrow \left(\mathsf{Blind}_1(\boldsymbol{k^A}[\sigma^e_{d_A}(i)], B_c^A)\right)_{i \in \mathbb{Z}_{N+1}}$

$D_A \leftarrow \mathsf{delay}\left(\mathsf{Enc}_{\boldsymbol{k^A}[0]}(s_A) \| \mathsf{H}(\boldsymbol{k^A}) \| \boldsymbol{B^A} \| d_A \| b_A\right)$

$\mathbf{S_1}:$ $\xrightarrow{D_A, \mathsf{H}\left(\mathsf{open}\left(D_A\right)\right), \boldsymbol{c^A}}$

$d_B \xleftarrow{\$} \mathbb{Z}_N; \boldsymbol{k^B} \xleftarrow{\$} \mathcal{K}^N$

$\boldsymbol{B^B} \xleftarrow{\$} \mathcal{BK}^{N+1}; B_c^B \xleftarrow{\$} \mathcal{BK}$

$\boldsymbol{c^B} \leftarrow \left(\mathsf{Blind}_1(\boldsymbol{k^B}[\sigma_{d_B}(i)], B_c^B)\right)_{i \in \mathbb{Z}_N}$

$\boldsymbol{d} \leftarrow \pi^{b_B}\left(\left(\mathsf{Blind}_2(\boldsymbol{c^A}[\sigma^e_{d_B}(i)], \boldsymbol{B^B}[i])\right)_{i \in \mathbb{Z}_{N+1}}\right)$

$D_B \leftarrow \mathsf{delay}\left(\mathsf{Enc}_{\boldsymbol{k^B}[0]}(s_B) \| \mathsf{H}(\boldsymbol{k^B}) \| \boldsymbol{B^B} \| d_B \| b_B\right)$

$\mathbf{S_2}:$ $\xleftarrow{D_B, \mathsf{H}\left(\mathsf{open}\left(D_B\right)\right), \boldsymbol{c^B}, \boldsymbol{d}}$

$\boldsymbol{c^{BA}} \leftarrow \left(\mathsf{Blind}_2(\boldsymbol{c^B}[\sigma_{d_A}(i)], \boldsymbol{B^A}[i])\right)_{i \in \mathbb{Z}_N}$

$\boldsymbol{e^B} \leftarrow \tau_N^{b_A}\left(\boldsymbol{c^{BA}}\right); \boldsymbol{e^A} \leftarrow \tau_{N+1}^{b_A}\left(\boldsymbol{d}\right)$

$\boldsymbol{m^A} \leftarrow \left(\mathsf{Unblind}_1(\boldsymbol{e^A}[i], B_c^A)\right)_{i \in \mathbb{Z}_{N+1}}$

$\mathbf{S_3}:$ $\xrightarrow{\boldsymbol{e^B}}$

$\boldsymbol{m^B} \leftarrow \left(\mathsf{Unblind}_1(\boldsymbol{e^B}[i], B_c^B)\right)_{i \in \mathbb{Z}_N}$

Exchange phase

$\mathbf{E_{2r+1}}:$ $\xrightarrow{\boldsymbol{m^A}[r]}$

$\mathbf{E_{2r+2}}:$ $\xleftarrow{\boldsymbol{m^B}[r]}$

Disclosure

$\mathbf{O_1}:$ $\xleftarrow{\mathsf{open}\left(D_B\right)}$

$\mathbf{O_2}:$ $\xrightarrow{\mathsf{open}\left(D_A\right)}$

**Fig. 4.** Main protocol

In the first two steps, we restrict the permutations to rotations. This makes the permutations commute without affecting the probability distribution of where the secrets are. After step 2, we obtain two lists where the secrets are in the same position. This in not enough to achieve optimality since the relative order of the secrets is fixed. If the messages were exchanged in their form after step 2, the secrets were in position $r$ and Alice were the first at sending the $r^{\text{th}}$ blinded value, then Bob's secret would follow Alice's. This would achieve a sub-optimal fairness probability of roughly $1 - \frac{1}{N}$.

The last two steps have the effect of shuffling the relative order of the secrets. If Bob places the extra dummy at the start of Alice's list, then he is effectively moving Alice's secret after his. Alice's action changes the relative order of any pair of messages. At the end of step 4, the relative order of the secrets is uniformly random and unknown to the parties. In particular, this represents the optimal distribution, i.e. no probability distribution of the 2 secret messages leads to a fairer exchange.

The last two messages $\mathbf{O_1}, \mathbf{O_2}$ are entirely optional and only used for computational optimisation, so that opening the delay encryption is made unnecessary. More specifically, they contain all the information needed to retrieve the secrets from the list of $\mathbf{E_i}$'s. Therefore, a party needs to undergo the expensive computation to open the delayed messages only if the exchange is not terminated or the other party actively cheated.

### 3.2 Security Requirements

After explaining the protocol, we can justify the need for the security parameters we set. The requirements on the delay encryption and symmetric encryption should be straight forward. These two primitives are only meant to hide the messages encrypted. The hash function is used as a commitment scheme, therefore the need for second preimage resistance. Collision resistance is not strictly needed but preimage resistance is required to avoid revealing something about the committed value.

The commutative blinding scheme must satisfy more complex requirements since it is the cornerstone of the setup phase. The scheme is required to hide the blinded values as well as hiding the shuffling. As a result, the simple COA requirement only ensures that nothing can be extracted from $\boldsymbol{c^A}$ or $\boldsymbol{c^B}$. The KPA requirement is needed if the protocol is aborted early before the actual secrets are exchanged. In this scenario, after the opening of the delay message, both parties can unblind the messages exchanged obtaining some of the dummy $\boldsymbol{k^A}$'s (or $\boldsymbol{k^B}$'s). Moreover, the delayed messages reveal the shuffling used, so a malicious party can perform a KPA on the values $\boldsymbol{c^A}$ (respectively $\boldsymbol{c^B}$).

The CPA requirements, using the set of permutations $\{\pi^b \circ \sigma_d \mid d \in \mathbb{Z}_N, b \in \{0,1\}\}$ (notation abuse), is needed to guarantee that Alice can't guess $d_B, b_B$ from $\boldsymbol{d}$. Similarly, the CPA requirement with permutations $\{\tau_N^b \circ \sigma_d \mid b \in \{0,1\}, d \in \mathbb{Z}_N\}$ prevents Bob from guessing $d_A, b_A$ from $\boldsymbol{e^B}$. It is worth pointing out that our choice of using rotations instead of arbitrary permutations results in stronger requirements for the commutative blinding. On the other side, arbitrary permutations do not commute, therefore the setup phase will need to be extended in order to keep the secrets next to each other.

Finally, the key indistinguishability requirement is used to guarantee that no information is leaked during the exchange phase. In particular, note that $\boldsymbol{m^B}[i]$ is blinded using a key which is chosen entirely upon on $b_A$. Therefore, Alice can unblind messages as she receives them. However, she can't check if the message received contain the correct key (since the encryption of the secret is delayed) and the KPA requirement prevents her from obtaining information from $\boldsymbol{c^B}$. On the other side, $\boldsymbol{m^A}[i]$ is blinded using a key dependent on $b_A$ and $b_B$. Therefore, Bob doesn't know a priori which key is used. If he were to discover which key is used to blind $\boldsymbol{m^A}[i]$, then he would discover $b_A$. This gives Bob knowledge of whether $\boldsymbol{k_A}[0]$ precedes or succeeds $\boldsymbol{k_B}[0]$, which is an unfair advantage.

### 3.3 Commutative Blinding and Why It Is Needed

The commutative blinding scheme is the most complex primitive used in our protocol. Quite a few options are available for commutative encryptions: ElGamal, SRA, Pohlig-Hellman, Massay-Omura, etc. However, most do not satisfy our requirements. For instance, ElGamal is not secure against our strong CPA requirement since each ciphertext is paired with a "tag" that encapsulates the random nonce used in the encryption process. On the other hand, we believe that the Pohlig-Hellman cipher satisfies our requirements, provided it is used carefully.[2]

It is important to note that all these cited commutative encryptions are based on the hardness of either the integer factorisation problem or the discrete logarithm problem. Since both are known to be vulnerable to quantum computers, the interesting question of finding post-quantum blindings arises. Most literature on post-quantum commutative encryption (e.g. [8,19]) is based on generalising the discrete logarithm problem to non-abelian groups. However, this approach may be flawed [20,27]. Other quantum resistant commutative encryption schemes might arise from group action on sets of isogenous elliptic curves as attempted by Stolbunov [26] or similarly to the CSIDH protocol [4]. As noted during the definition of the commutative blinding, we believe homomorphic encryption could be used to construct this primitive. The most trivial implementation would require the outer blinding $\mathsf{Blind}_1$ to be the homomorphic encryption, while the inner blinding $\mathsf{Blind}_2$ can be a simple Vernam cipher. Since $\mathsf{Blind}_2$ is always used with unique keys, the overall construction's security should follow from the security of the homomorphic encryption. In this regard, we wish to point out that most fully homomorphic encryption schemes rely on the hardness of the Learning With Errors problem (or its ring variant) which was proved to be at least as hard as some worst case lattice problems [22] such as the Shortest Vector Problem. As

[2] In particular, we need to prevent the ciphertexts from being recognised from their order. Thus, the modulus picked should be a safe prime $p = 2q + 1$ and all plaintexts should be forced to be in the same subgroup of prime order $q$.

a result, these schemes are good candidates for building quantum-safe blinding schemes.

Given the complexity of commutative blinding, a natural question is whether we can achieve the same results without its use. Therefore, we feel the need to briefly justify its use. Our problem of hiding the secret messages among a list of other dummy messages is similar to the issue of shuffling a deck of cards for an online game of poker. No single party must know the whole shuffle and both need to contribute to it. This "mental poker" problem is often solved using commutative blinding, oblivious transfer or by generating cards on the fly. [12–14,25] The last option is clearly not an option in our context. On the other hand, the use of oblivious transfer is appealing yet we believe this is only possible (without greatly increasing the round complexity of the protocol) if the shuffling phase can be split into two sub-shuffles each performed independently by the two parties. We could not find such a split, therefore we consider the existence an efficient setup phase without commutative blinding an open question.

## 4 Analysis

A protocol $\Pi^{M+1}$ is a sequence of messages $\langle m_0, \ldots, m_M \rangle$. A time point $t$ in the protocol $\Pi^M$ is an integer $t \in \mathbb{Z}_M$. We say that messages $m_i$ with $i \leq t$ happens before $t$. We write $T_A$ (and $T_B$) for the time point after which Alice (respectively Bob) is guaranteed to obtain $s_B$ (respectively $s_A$). For instance, $m_0 = \mathsf{Enc}_k(s_A) \wedge m_3 = k \implies T_B \leq 3$. A subtler example is: $m_0 = \mathsf{Enc}_k(s_A) \wedge m_4 = \mathsf{delay}(k) \implies T_B \leq 4$. Using this simple model, we can analyse the fairness of our protocol against passive adversaries. In this context, we define a passive adversary as one that is only allowed to go offline unexpectedly.

In this section, we write $s_A \in \boldsymbol{m^A}[i]$ to mean that $\boldsymbol{m^A}[i]$ is the blinding of $\boldsymbol{k^A}[0]$. Essentially, $\boldsymbol{m^A}[i]$ is the only important message in $\boldsymbol{m^A}$. The analogue notation $s_B \in \boldsymbol{m^B}[i]$ will also be used.

### 4.1 Adversarial Model

In our adversarial model, the enemy can control Alice or Bob (but not both) and they are allowed only to perform actions that do not reveal them as a bad actor. We define this kind of adversary as a *covert adversary.* We assume communications channels are authenticated, guarantee integrity and confidentiality and are secure against reordering and replay attacks. Therefore receiving a malformed message signed by the adversary constitute a proof of their misbehaviour, however, aborting communication is not assumed to be a malicious action. To prove the security of our protocol, we first show that it is secure against *passive* adversaries which are only allowed to perform side computations and abort communication. We then show that any covert adversary is restricted to behave like a passive adversary.

### 4.2 Fairness

We will show that the protocol of Fig. 4 achieves $\frac{1}{2N}$-partial fairness. The key idea is that $T_A$ and $T_B$ are kept next to each other so that there is only one successful early-termination attack among the $2N$ possible. Due to the use of delay encryption, we see that $T_A = i + 2$ if and only if $E_i$ is the blinding of $\boldsymbol{k}^{\boldsymbol{B}}[0]$. Similarly, $T_B = i + 2$ if and only if $E_i$ is the blinded $\boldsymbol{k}^{\boldsymbol{A}}[0]$. Therefore, we compute the distribution of $T_A, T_B$ by looking at the distribution of $\boldsymbol{k}^{\boldsymbol{A}}[0]$ and $\boldsymbol{k}^{\boldsymbol{B}}[0]$.

**Theorem 1.** *Let $\Pi$ be the fair exchange protocol from Fig. 4. Then*

$$\Pr\left[s_A \in \boldsymbol{m}^{\boldsymbol{A}}[i] \wedge s_B \in \boldsymbol{m}^{\boldsymbol{B}}[j]\right] = \begin{cases} \frac{1}{2N} & \textit{if } i = j \vee i = j + 1 \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

*Proof.* Note that

$$\begin{aligned} \boldsymbol{m}^{\boldsymbol{A}}[i] = \mathsf{Blind}_2 \Big( & \boldsymbol{k}^{\boldsymbol{A}}[\sigma^e_{d_A+d_B}((i(-1)^{b_A} + Nb_A - b_B) \bmod (N+1))], \\ & \boldsymbol{B}^{\boldsymbol{B}}[(i(-1)^{b_A} + Nb_A - b_B) \bmod (N+1)]\Big) \end{aligned} \tag{2}$$

$$\begin{aligned} \boldsymbol{m}^{\boldsymbol{B}}[i] = \mathsf{Blind}_2 \Big( & \boldsymbol{k}^{\boldsymbol{B}}[(i(-1)^{b_A} + (N-1)b_A + d_A + d_B) \bmod N], \\ & \boldsymbol{B}^{\boldsymbol{A}}[i(-1)^{b_A} + (N-1)b_A]\Big) \end{aligned} \tag{3}$$

So

$$\begin{aligned} s_A \in \boldsymbol{m}^{\boldsymbol{A}}[i] &\iff 0 = \sigma^e_{d_A+d_B}((i(-1)^{b_A} + Nb_A - b_B) \bmod (N+1)) \\ s_B \in \boldsymbol{m}^{\boldsymbol{B}}[i] &\iff 0 = (i(-1)^{b_A} + (N-1)b_A + d_A + d_B) \bmod N \end{aligned}$$

Hence

$$\begin{aligned} \Pr\left[s_B \in \boldsymbol{m}^{\boldsymbol{B}}[i]\right] &= \Pr_{\substack{b_a \xleftarrow{\$} \{0,1\} \\ d_A, d_B \xleftarrow{\$} \mathbb{Z}_N}} \left[0 = (i(-1)^{b_A} + (N-1)b_A + d_A + d_B) \bmod N\right] \\ &= \frac{1}{2}\Big( \Pr_{d \xleftarrow{\$} \mathbb{Z}_N} [0 = (i + d) \bmod N] + \\ &\qquad \Pr_{d \xleftarrow{\$} \mathbb{Z}_N} [0 = (N - 1 - i + d) \bmod N] \Big) = \frac{1}{N} \end{aligned}$$

Assume $s_B \in \boldsymbol{m}^{\boldsymbol{B}}[j]$, that is:

$$0 = (j(-1)^{b_A} + (N-1)b_A + d_A + d_B) \bmod N$$

We will show that $s_A$ is either in $\boldsymbol{m}^{\boldsymbol{A}}[j]$ or $\boldsymbol{m}^{\boldsymbol{A}}[j+1]$ by analysing the four cases $(b_A, b_B) \in \{0, 1\}^2$

1. $\mathbf{b_A = b_B = 0}$

$$s_B \in \boldsymbol{m^B}[j] \implies \sigma^e_{d_A+d_B}(j) = (j + d_A + d_B) \bmod N = 0 \implies s_A \in \boldsymbol{m^A}[j]$$

2. $\mathbf{b_A = b_B = 1}$

$$s_B \in \boldsymbol{m^B}[j] \implies \sigma^e_{d_A+d_B}(N-1-j) = (N-1-j+d_A+d_B) \bmod N = 0 \\ \implies s_A \in \boldsymbol{m^A}[j]$$

3. $\mathbf{b_A = 0 \wedge b_B = 1}$

$$s_B \in \boldsymbol{m^B}[j] \implies \sigma^e_{d_A+d_B}(j+1-b_B) = (j + d_A + d_B) \bmod N = 0 \\ \implies s_A \in \boldsymbol{m^A}[j+1]$$

4. $\mathbf{b_A = 1 \wedge b_B = 0}$

$$s_B \in \boldsymbol{m^B}[j] \implies \\ \sigma^e_{d_A+d_B}(N-(j+1)) = (N-j-1+d_A+d_B) \bmod N = 0 \\ \implies s_A \in \boldsymbol{m^A}[j+1]$$

As a result,

$$\Pr_{b_A,b_B \xleftarrow{\$} \{0,1\}}\left[s_A \in \boldsymbol{m^A}[i] \,\middle|\, s_B \in \boldsymbol{m^B}[j]\right] = \begin{cases} \frac{1}{2} & \text{if } i = j \vee i = j+1 \\ 0 & \text{otherwise} \end{cases}$$

The statement of the theorem follows directly from the equation

$$\Pr\left[s_A \in \boldsymbol{m^A}[i] \wedge s_B \in \boldsymbol{m^B}[j]\right] = \\ \Pr\left[s_A \in \boldsymbol{m^A}[i] \,\middle|\, s_B \in \boldsymbol{m^B}[j]\right] \Pr\left[s_B \in \boldsymbol{m^B}[j]\right]$$

□

**Corollary 1.** *Let $\Pi$ be the protocol from Fig. 4. Let $\mathcal{A}$ be a computationally bounded passive adversary. Then $\mathcal{A}$ can successfully obtain the other party's secret without revealing theirs with probability $\frac{1}{2N}$.*

*Proof (Sketch).* First, recall Eq. 1. All we have to prove is that $\mathcal{A}$ cannot obtain any knowledge that gives them any better probability distribution. The parameters $\mathfrak{p}$ which determines the distributions are $d_A, d_B, b_A, b_B$. In Sect. 3.2, we have described what role each security requirement plays in guaranteeing the fairness of the protocol, but here we report briefly the key points.

Note that message $\mathbf{S_1}, \mathbf{S_2}, \mathbf{S_3}$ don't reveal anything about $\mathfrak{p}$ until the delays open *after* the protocol termination. This is given by the properties: delay-COA, H-preimage resistance, $\mathsf{Blind_1}$-COA and $\mathsf{Blind_2}$-CPA. The messages $\mathbf{E_{2r-1}}$ and $\mathbf{E_{2r}}$ do not reveal anything about $\mathfrak{p}$ since $\mathsf{Blind_2}$ is key-indistinguishable and $\mathsf{Blind_1}$ is KPA secure. Therefore, $\mathcal{A}$ only knows the parameters they picked and these are not enough to skew the probability distribution of Eq. 1.

Say that $\mathcal{A}$ stops after message $\mathbf{S_i}$, then they will obtain the other party's secret with probability 0. Say that $\mathcal{A}$ stops after message $\mathbf{E_i}$, then they will successfully cheat if $\mathbf{E_i}$ contains the other party's secret and $s_{\mathcal{A}} \in \mathbf{E_j}$ with $j > i$. By Eq. 1 this happens with probability $\frac{1}{2N}$. Stopping during the disclosure phase is pointless as $s_{\mathcal{A}}$ is already transmitted. □

### 4.3 Optimality

In the protocol from Fig. 4, we hide the secrets $s_A, s_B$ among the messages $\mathbf{E_i}$'s and use delay encryption to guarantee that $T_A = i + 2 \iff s_B \in \mathbf{E_i}$ (and similarly for Bob). Note that the +2 is needed to account for the setup phase. Hence, $T_A$ and $T_B$ are also hidden for the entire duration of the exchange phase.

Here, we prove that our construction is "optimal", meaning that no exchange phase of $M$ messages can achieve fairness greater than $1 - \frac{1}{M-1}$. However, this leaves open the question of whether our setup phase can be shortened. In this regard, we point out that messages $\mathbf{S_3}$ and $\mathbf{E_1}$ can be sent as one. Therefore, the proposed protocol achieves $\frac{1}{M-1}$-partial fairness using $M + 2$ messages.

**Lemma 1.** *$A$ receives a message at time $T_A$.*

*Proof.* Assume for a contradiction that $m_{T_A}$ is a message from Alice to Bob. It follows that Alice can compute $m_{T_A}$ from $\{m_1, \dots, m_{T_A - 1}\}$. As a result, everything that Alice can compute from $\{m_1, \dots, m_{T_A}\}$ can be done from $\{m_1, \dots, m_{T_A - 1}\}$. Therefore $T_A$ is not minimal. □

**Lemma 2.** *In the absence of third parties, $T_A \neq T_B$.*

*Proof.* This follows directly from Lemma 1. □

**Lemma 3.** *Assume there is no third party. If in $\Pi$ messages do not alternate between Alice and Bob, then there is another protocol $\Pi'$ which is as fair as $\Pi$ in which the messages alternate.*

*Proof.* Let $\Pi$ be a protocol where the messages between Alice and Bob do not alternate. Construct $\Pi'$ from $\Pi$ by collapsing consecutive messages from the same party into one. After this process, append "dummy" messages so that $\Pi'$ and $\Pi$ have the same length. In this process, the fairness of $\Pi$ is left untouched. In particular, note that the optimal strategy of any adversary could have not been to stop between consecutive messages and it is definitely not to stop after dummy messages. □

**Theorem 2.** *Let $\Pi$ be a fair exchange protocol between Alice and Bob in absence of third parties. If $\Pi$ consists of $M + 1$ messages, then there is an unfair run with probability $\frac{1}{M}$*

*Proof.* By Lemma 3 we can assume that $\Pi = \langle m_0, \dots, m_M \rangle$ where Alice sends the messages with even index, and Bob those with odd index. There are $m$ possible attacks on the protocol by early abortion: stop the protocol after $i$ messages where $0 < i \leq M$. By Lemma 2 we know that at least one of these attacks would be successful. So, the probability of an unfair run is at least $\frac{1}{M}$. □

### 4.4 Catching Active Cheaters

So far we have analysed the fairness of our protocol when the adversary is passive. We now look at the case where the adversary can alter the messages they send. Our aim is to show that any such adversary will be caught. That is, our protocol allows the honest party to demonstrate the misbehaviour of the other party. Since channels are authenticated a transcript of the protocol constitute a proof of misbehaviour. Hence, covert adversary can only behave like passive adversaries.

**Theorem 3.** *If a party misbehaves, the other can prove it (unless both misbehaved).*

*Proof.* Firstly, note that inconsistencies between the delayed messages and the shuffling “sub-messages" $\boldsymbol{c^A}, \boldsymbol{c^B}, \boldsymbol{d}, \boldsymbol{e^B}$ will result in inconsistent $\mathbf{E_i}$'s. We assume that Alice and Bob have agreed on predicates (i.e. boolean functions) $\mathsf{Exp}_A, \mathsf{Exp}_B$ to indicate what they expect to receive. Recall Eqs. 2 and 3 and assume that both parties have knowledge of the content of the delayed messages $D_A, D_B$, which they eventually will. Therefore, both parties have complete knowledge of the permutations used. Each exchange messages $\mathbf{E_i}$ can be unblinded and reordered so that they are expected to match $\mathsf{H}(\boldsymbol{k^A}[j])$ (or $\mathsf{H}(\boldsymbol{k^B}[j])$) for some $j$. Finally, the secrets can be retrieved and checked against $\mathsf{Exp}_A$ or $\mathsf{Exp}_B$. Assume that $\boldsymbol{m^A}[i]$ is not correct, then Bob could be responsible only if they have wrongly computed $\boldsymbol{d}$. However, note that the function $\boldsymbol{c^A} \mapsto \boldsymbol{d}$ is entirely determined by $(d_B, b_B, \boldsymbol{B^B})$. Therefore, one can verify the computation of $\boldsymbol{d}$ and, if this is correct, prove that Alice was responsible for the issue with $\boldsymbol{m^A}[i]$. Similarly, Alice's computation $\boldsymbol{c^B} \mapsto \boldsymbol{e^B}$ is determined by $(d_A, b_A, \boldsymbol{B^A})$. It follows that anyone holding a transcript of the protocol (and $\mathsf{Exp}_A, \mathsf{Exp}_B$) can check if each $\mathbf{E_i}$ is correct and determine who introduced any eventual error. □

## 5 Conclusion and Future Research

In this paper, we have presented a fair exchange protocol which achieves fairness against covert adversaries with probability $1 - \frac{1}{2N}$ in $2N + 1$ exchange messages and 3 setup messages. We proved that the protocol is optimally fair up to shortening the already-short setup phase. This impossibility result holds in a general model which only assumes that the involved parties are not computationally unbounded. The cryptographic primitives that allow the design of the optimally-fair protocol are delay encryption and commutative blinding. The use of delay encryption introduces the reasonable assumption that both parties have somewhat similar sequential computational power, a considerable improvement on the results of [7] where similar (parallel) computing power was needed. Nevertheless, we limited the use of delay encryption to a single message which will be opened only when a party suspects misbehaviour. This gives some leeway to use less efficient algorithms. We showed that our protocol is secure against covert adversaries without the need of expensive constructions and point out that all the primitive used can be quantum safe. We overcome the bounds of [16] by extending the cryptographic palette and achieve a stronger fairness than [6] by careful choice of the blinding strategy. Despite the substantial contribution of our paper, more research is needed in the field of fair exchange. The only drawback of our protocol is the need for a strong commutative blinding. Therefore, it raises the interesting question of whether our protocol can be modified so as not to use commutative encryption without increasing the message complexity of the setup phase. On the more practical side, an interesting problem to consider is the scenario where the malicious party aborts the exchange and, if the secrets are not exchanged, they begin another exchange with the same party and same secrets. This seems a plausible scenario in real-world applications since the honest party is likely to need the exchange to succeed, and they will try again if network failures seem the cause of abortion. However, the fairness of the protocol is different since multiple runs of the same protocol must be taken into account.

## References

1. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for multi-party fair exchange. IBM Research Division (1996)
2. Ben-Or, M., Goldreich, O., Micali, S., Rivest, R.L.: A fair protocol for signing contracts. IEEE Trans. Inf. Theory **36**(1), 40–46 (1990). https://doi.org/10.1109/18.50372
3. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_15
4. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11274, pp. 395–427. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03332-3_15
5. Cleve, R.: Limits on the security of coin flips when half the processors are faulty. In: Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing (STOC 1986), pp. 364–369. Association for Computing Machinery, New York (1986). https://doi.org/10.1145/12130.12168
6. Couteau, G., Roscoe, A.W., Ryan, P.Y.A.: Partially-fair computation from timed-release encryption and oblivious transfer. In: Baek, J., Ruj, S. (eds.) ACISP 2021. LNCS, vol. 13083, pp. 330–349. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90567-5_17
7. Damgård, I.B.: Practical and provably secure release of a secret and exchange of signatures. J. Cryptol. **8**(4), 201–222 (1995). https://doi.org/10.1007/BF00191356
8. Dmitriy, M.: Non-commutative finite groups as primitive of public key cryptosystems. Quasigroups Related Syst. **18**(2), 165–176 (2010)
9. Even, S., Yacobi, Y.: Relations among public key signature systems. Report, TECHNION - Israel Istitute of Technology (1980)
10. Franklin, M., Tsudik, G.: Secure group barter: multi-party fair exchange with semi-trusted neutral parties. In: Hirchfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 90–102. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055475
11. Garbinato, B., Rickebusch, I.: Impossibility results on fair exchange. In: Eichler, G., Kropf, P., Lechner, U., Meesad, P., Unger, H. (eds.) 10th International Conference on Innovative Internet Community Systems (I2CS) (Jubilee Edition 2010), pp. 507–518. Gesellschaft für Informatik e.V, Bonn (2010)

12. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC 1987), pp. 218–229. Association for Computing Machinery, New York (1987). https://doi.org/10.1145/28395.28420
13. Goldwasser, S., Micali, S.: Probabilistic encryption & how to play mental poker keeping secret all partial information. In: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing (STOC 1982), pp. 365–377. Association for Computing Machinery, New York (1982). https://doi.org/10.1145/800070.802212
14. Golle, P.: Dealing cards in poker games. In: International Conference on Information Technology: Coding and Computing (ITCC 2005) - Volume II, vol. 1, pp. 506–511 (2005). https://doi.org/10.1109/ITCC.2005.119
15. Gollmann, D.: Computer Security. Wiley Textbooks, Chichester (2011)
16. Gordon, S.D., Katz, J.: Partial fairness in secure two-party computation. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 157–176. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_8
17. Kaufman, C., Perlman, R., Speciner, M.: Network Security: Private Communication in a Public World, 2nd edn [electronic resource]. Prentice Hall PTR (2002)
18. Maffei, I., Roscoe, A.W.: Delay encryption by cubing (2022). https://doi.org/10.48550/ARXIV.2205.05594
19. Moldovyan, A.A., Moldovyan, D.N., Moldovyan, N.A.: Post-quantum commutative encryption algorithm. Comput. Sci. J. Moldova **27**, 299–317 (2019)
20. Myasnikov, A.D., Ushakov, A.: Quantum algorithm for the discrete logarithm problem for matrices over finite group rings. Cryptology ePrint Archive, Report 2012/574 (2012)
21. Pagnia, H., Gärtner, F.C.: On the impossibility of fair exchange without a trusted third party. Darmstadt University of Technology, Technical Report (1999)
22. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing (STOC 2005), pp. 84–93. Association for Computing Machinery, New York (2005). https://doi.org/10.1145/1060590.1060603
23. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-Lock Puzzles and Timed-Release Crypto. Report, Massachusetts Institute of Technology (1996)
24. Roscoe, A.W., Ryan, P.Y.A.: Auditable PAKEs: approaching fair exchange without a TTP. In: Stajano, F., Anderson, J., Christianson, B., Matyáš, V. (eds.) Security Protocols 2017. LNCS, vol. 10476, pp. 278–297. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71075-4_31
25. Shamir, A., Rivest, R.L., Adleman, L.M.: Mental poker. In: The Mathematical Gardner, pp. 37–43. Springer, Boston (1981). https://doi.org/10.1007/978-1-4684-6686-7_5
26. Stolbunov, A.: Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. Adv. Math. Commun. **4**(2), 215–235 (2010)
27. Yanlong, M.: Cryptanalysis of the cryptosystems based on the generalized hidden discrete logarithm problem. Cryptology ePrint Archive, Report 2021/1701 (2021)