

Two Party Fair Exchange

Abstract. Fair Exchange (FE) protocols are a class of cryptographic protocol in which two parties, X and Y , exchange some secret data, where the ability of each party to receive secret data is contingent on having sent secret data of their own. When exchanging secret data without the support of FE protocols, whoever sends their secret first makes themselves vulnerable to the possibility that the other participant will cheat and won't send their secret in return. It is widely believed that FE protocols satisfying the soundness notion of *Strong Fairness* require a third party arbitrator trusted by both X and Y , however in this paper we construct an FE scheme satisfying a notion of *Strong Fairness* with no third party involvement. This is achieved by embracing non-determinism and slightly relaxing the definitions of security from the perfect to the computational domain. The protocol presented in this paper allows two parties running interactive processes to exchange secrets, safe in the knowledge that either both secrets will change hands, or neither will, without the involvement of a trusted third party. The security of the protocol presented in this work reduces to the strength of an underlying symmetric authenticated encryption scheme.

1 Introduction

It is not uncommon for two mutually distrusting parties to wish to exchange some secret information. These types of transactions occur frequently in the wild: economic trades where both parties might exchange a cryptographic signature approving the transfer of funds[17]; swapping gossip between peers; trading the output of executed programs; or providing payment for computational work conducted in the cloud. Similarly, corporations or nation state allies may wish to trade sensitive strategic information. Some cryptographic protocols even end with the assumption that an untrusted party will dutifully pass a result to the other. For example, in Multi Party Computation (MPC) constructions[19] the circuit garbler is expected to dutifully reveal the decrypted results of joint calculations to the circuit evaluator. In some untrusted settings this MPC construction would be greatly improved by using an FE protocol to exchange the garbled circuit output for the keys used to decrypt it.

Currently, this problem is solved by using a third party to mediate the transaction. This third party might be a bank, mutual friend, or some blockchain-backed protocol[10,12,7]. In the naive case, the third party holds both secrets until they have been validated, and then forwards the validated secrets to each party. Trusted third parties are used ubiquitously in modern FE protocols, which aim to improve over the naive case by reducing the involvement of a third party as much as possible, allowing two parties to exchange secrets with minimal

interference[1]. Current state-of-the-art FE protocols require an offline trusted authority that is only required to intervene in the case of a dispute[2,14], this approach is known as an *Optimistic* FE protocol as the third party is only involved if one participant acts dishonestly. Reducing the involvement of the third party is one of the key objectives of FE protocol research. It's widely believed that a FE construction is not possible without a trusted third party performing arbitration[18].

In this work we present a two party FE protocol with security that reduces to the strength of an underlying IND-CPA secure authenticated encryption scheme. To the best of our knowledge this is the first FE protocol that satisfies a notion of *Strong Fairness* without the involvement of a trusted third party. The key insight resulting in this breakthrough construction is that the impossibility result for two party FE with *Strong Fairness* only holds for a deterministic protocol where the *Strong Fairness* definition is interpreted as a definition of soundness in the domain of perfect security[18]. This result can be circumvented by embracing a constant configurable probability of failure, and by relaxing *FE* definitions from the domain of perfect security to computational security.

1.1 Contributions

In this work we:

- Relax standard FE definitions to a more achievable notion of computational security.
- Define a two party FE scheme with security reducing to the strength of an IND-CPA secure symmetric authenticated encryption scheme.
- Prove the scheme has *Computational Completeness* and *Computational Strong Fairness* properties, as per the newly-relaxed FE definitions.

1.2 Assumptions

In order to simplify the constructions, it is assumed that all parties send a secret which satisfies some property expected by the receiving party. For example, the property could be that the secret is a valid signature on a predetermined message.

Assumption 1 *There exists a function $\text{Desc}(d)$ that provides a description of data d . Both participants in the scheme agree on the description of the data they plan to send and receive in advance of the FE protocol. Both participants send valid encryptions of their agreed upon data at the start of the protocol and do not, for example, swap d for random noise.*

This can be enforced with **ZKP!**s (**ZKP!**s) operating in parallel to the FE protocols in this work. Furthermore, we assume all parties will follow the protocol honestly, up to the moment where one secret is revealed, at which point a malicious party will do what they can to avoid revealing their secret in return.

Assumption 2 *The protocol will be followed honestly, the only deviations allowed by an adversary are deviations by omission, in which an adversary can decide to send no data.*

As before, this can be enforced with **ZKP!**s and commitment schemes operating in parallel to the protocol. These assumptions were made as they make the protocol definitions that follow simpler and are reasonable assumptions that can be enforced with appropriate measures by any implementers.

The standard assumption that there exists an IND-CPA secure symmetric authenticated encryption scheme is also used in this work. This encryption scheme is used as a black box in the presented FE constructions, the interface to which is described in the Background section of this chapter.

Assumption 3 *There exists an IND-CPA secure symmetric authenticated encryption scheme.*

The third and final protocol presented in this chapter makes use of a conceivable but non-standard assumption; that there is a “secure” λ and an “insecure” λ' such that the adversarial advantage against each is separated by a factor M , and M is independent of λ' .

Assumption 4 *There exists security parameter λ , and λ' , and constant M such that*

$$M \mathsf{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda') \quad (1)$$

where M is independent of λ' and λ is considered secure, but λ' is considered bruteforceable.

There is an obvious and direct relationship between the adversarial advantage and the time one would expect it to take to mount a bruteforce attack on an instantiation of a cryptosystem, a fact which we use to demonstrate the reasonableness of this assumption. Consider an instantiation of a cryptosystem deemed “insecure” for a given domain, $\mathcal{E}(\lambda')$, with a key space bruteforceable within one month. There exists a corresponding instantiation $\mathcal{E}(\lambda)$ with a key space bruteforceable in 1000 years. In this example $M = 12,000$ and is fixed, no matter if new algorithmic developments reduce the time taken to bruteforce the key space. In such a situation, where λ and λ' need to be updated because of algorithmic advances or hardware improvements in order to maintain their expected time-to-bruteforce of 1000 years and 1 month respectively, M is clearly unaffected.

In conventional security practice the security parameter is chosen to satisfy some impractical time required to mount a successful attack on the system. As attack methods are developed, λ is adaptively increased. It is easy to see that M is therefore not directly related to λ or λ' and is instead dependent on the relationship between the two. No matter how M is chosen, it is constant as λ and λ' adaptively grow.

As a concrete example, consider two corresponding instantiation of AES, one “secure” and one “bruteforceable”, in the face of an adversary with access to

common commodity computer hardware. Suppose a “secure” instantiation of AES is selected, one with an expected time-to-attack of 1000 years. It is worth noting here that security practitioners tend to aim for a more existential time-to-attack, in the order of the heat death of the universe. The definitions relating to what is secure enough vary from setting to setting depending on the relative strengths of expected adversaries. However, in FE the goal is for targets to eventually exchange secrets, so a key space attackable in 1000 years on commodity hardware could very well be considered adequately secure. With some back of the envelope calculations an AES instantiation with an expected time-to-attack of 1000 years would result in a key length of $\lambda \approx 84$ bits. The complimentary “bruteforceable” instantiation, with key space attackable in a month, would have a key length of $\lambda' \approx 70$ bits. In this case we have $M \approx 2^{14}$. Note that as we adaptively change λ and λ' to stay ahead of algorithmic improvements M will stay fixed.

This final protocol presented in this chapter also requires that the computational capabilities of one of the two participants is publicly and reliably known to within an order of magnitude.

Assumption 5 *The computational capabilities of party X are publicly and reliably known to within an order of magnitude.*

1.3 Notation

In pseudocode \leftarrow indicates assignment. We make use of standard cryptographic notation: k is a cryptographic key, encryption schemes are defined as a set of functions $\mathcal{E} = (\text{KGen}, \text{Enc}, \text{Dec})$ and λ is a security parameter. Functions that grow negligibly in the security parameter λ are denoted with $\text{negl}(\lambda)$. At times we make use of vectors to indicate an ordered set of data, these are denoted by a variable in bold \mathbf{v} . The i th element of the vector \mathbf{v} is denoted by $\mathbf{v}[i]$. $[x; n]$ is used as shorthand for an n length vector with the value x at every position. $\mathcal{E}(\lambda)$ is used as shorthand to indicate the encryption scheme \mathcal{E} used with security parameter λ during KGen.

We make black-box use of 1-in- n oblivious transfer[5]. Oblivious transfer allows a Sender to offer up a vector of data, \mathbf{y} , and a Receiver is able to access a single element indexed at i without the Sender learning which element the Receiver accessed, and without the Receiver learning any of the unaccessed data in the vector \mathbf{y} . The use of oblivious transfer is noted with $x \xleftarrow{\text{OT}} \mathbf{y}[i]$, where x is the local variable obtained by the Receiver, \mathbf{y} is an ordered vector of data stored on the Sender, and i is the index in \mathbf{y} accessed as part of the oblivious transfer.

The two participants in our FE scheme are referenced as party X and party Y . Variables associated with either party are usually subscripted with the appropriate letter. In the protocol description “ $X :$ ” is a prefix used to indicate the operation is executed by party X . Party X learning secret d_Y is denoted with the shorthand $X(d_Y)$.

2 Background

Many references to underlying cryptographic building blocks are kept intentionally abstract. In this section we define the properties made use of later in this work.

2.1 Fair Exchange

These definitions are adapted from [14]. In a FE protocol party X holds one piece of data, d_X , while party Y holds a different piece of data, d_Y . The protocol is *Complete* if, with honest parties, when the protocol terminates X has learned d_Y and Y has learned d_X .

Definition 1 (Complete). *With honest parties X and Y , and respective data d_X and d_Y , when the protocol terminates*

$$X(d_Y) \wedge Y(d_X) \tag{2}$$

The difficulty arises in attempting to construct a protocol in which a dishonest participant is not able to discover the secret of the other participant without giving up their own secret. There should be no way of X learning anything about d_Y without offering up d_X to Y . The same is true for Y in relation to X 's secret, d_X . This property is *Strong Fairness*.

Definition 2 (Strong Fairness). *When the protocol terminates, either $X(d_Y)$ or Y has gained no information about d_X . Similarly, when the protocol has completed either $Y(d_X)$ or X has gained no information about d_Y .*

The *Strong Fairness* property contrasts with the type of fairness provided by early FE schemes. Earlier schemes usually operated by gradually disclosing a secret over many iterative rounds[16]. Obviously, with this approach a cheating participant can terminate at any point and have learned some information about the honest participant's secret. Fair exchange schemes featuring gradual disclosure, along with other related works, are discussed further in Section 5.

A naive solution which satisfies *Strong Fairness* is to allow all messages to pass through a trusted third party, A . This approach is shown in Figure 1. The current state-of-the-art only requires the trusted third party to intervene in the case of a dispute[14].

It's widely believed that strong fairness is impossible without the use of a trusted third party[18]. This belief stems from a result in which two party FE protocols can be used to construct a deterministic asynchronous consensus scheme, and as there are no deterministic asynchronous algorithms for consensus[9], there can be no deterministic two-party FE protocols with *Strong Fairness*.

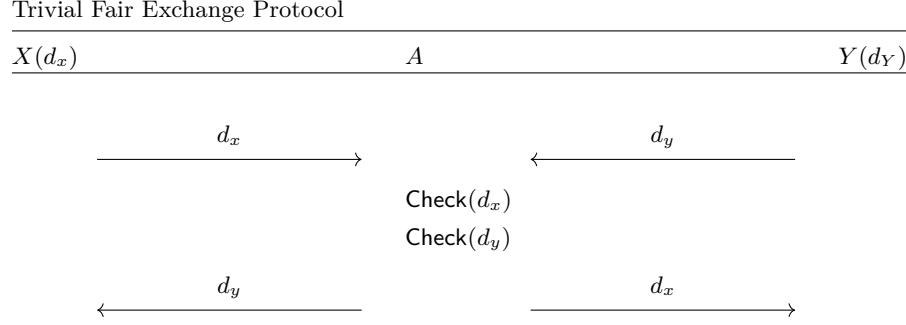


Fig. 1: Trivial FE protocol with a trusted third party arbitrator, A .

2.2 Symmetric IND-CPA Secure Authenticated Encryption Schemes

A symmetric encryption scheme $\mathcal{E} = (\text{KGen}, \text{Enc}, \text{Dec})$ is an authenticated encryption scheme if, along with confidentiality of data, it guarantees data integrity and authenticity. An authenticated encryption scheme is defined by the following functions:

- **Key Generation:** $\mathbf{k} \leftarrow \text{KGen}(\lambda)$; generate key \mathbf{k} .
- **Encryption:** $c \leftarrow \text{Enc}_{\mathbf{k}}(m)$; encrypt message m to produce ciphertext c .
- **Authenticated Decryption:** $m \text{ or } \perp \leftarrow \text{Dec}_{\mathbf{k}}(c)$; decrypt ciphertext c to produce message m , with an invalid combination of c and \mathbf{k} decrypting to \perp .

In the case a ciphertext has been tampered with, or does not originate from the expected sender Dec outputs \perp . Authenticated encryption schemes are usually composed from encryption schemes and message authentication codes.

A scheme is IND-CPA secure if it is indistinguishable under a chosen plaintext attack[3]. Consider the security game detailed in Figure 2 and the following definition.

Definition 3 (Symmetric IND-CPA Secure Authenticated Encryption Scheme). Consider symmetric authenticated encryption scheme $\mathcal{E} = (\text{KGen}, \text{Enc}, \text{Dec})$. The advantage of an adversary \mathcal{A} in the IND-CCA security game, detailed in Figure 2, is defined as:

$$\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) = 2 \Pr[\text{INDCPAGame}(\lambda) = 1] - 1 \quad (3)$$

A cryptosystem \mathcal{E} is IND-CPA secure if for any probabilistic polynomial time adversary \mathcal{A}

$$\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \leq \text{negl}(\lambda) \quad (4)$$

INDCPAGame₁(λ)
1 : $k \leftarrow KGen(\lambda), b \leftarrow \{0, 1\}$
2 : $m_0, m_1 \leftarrow \mathcal{A}^{OEnc_k}()$
3 : $c \leftarrow Enc_k(m_b)$
4 : $b' \leftarrow \mathcal{A}^{OEnc_k}(c)$
5 : if $b = b'$ then return 1
6 : else return 0

Fig. 2: IND-CPA security game. The adversary wins if they can differentiate an encryption of one of two messages of their choice. They are given access to an encryption oracle $OEnc_k$ which is able to provide valid encryptions interactively to the adversary. There is an implicit adversary state maintained between the two calls to the adversary \mathcal{A} .

3 Extended Definitions for Fair Exchange

In this section we define new slightly relaxed variations of standard FE definitions, which are more suitable to the work presented in this paper. The new definitions adapt existing FE definitions from the domain of perfect security to computational security. Although the new definitions presented in this section provide a weaker notion of security than those provided in some existing work, they are much stronger than the historical definitions of security used by older FE protocols built on the principle of gradual disclosure.

As an extension to the FE notion of *Completeness* we provide a definition for *Computational Completeness*.

Definition 4 (Computational Completeness). *Given party X with data d_X and party Y with data d_Y . If both X and Y act honestly, X learns d_Y and Y learns d_X with probability*

$$\Pr[X(d_Y) \wedge Y(d_X)] \approx 1 \quad (5)$$

Similarly, we provide a definition for *Computational Strong Fairness*. This definition uses the security game detailed in Figure 3 to simulate the ability of one of the two parties in the protocol to act maliciously. Importantly, the security game is defined in terms of parties P and Q in order to separate them from the concrete roles of X and Y in our FE protocol definition.

Definition 5 (Computational Strong Fairness). *For arbitrary party P define the advantage in the PFairnessGame defined in Figure 3 as*

$$\text{Adv}_{\mathcal{A}_P}^{\text{fairness}}(\delta, \lambda) = \Pr[\text{PFairnessGame}(\delta, \lambda) = 1] \quad (6)$$

where PFairnessGame is defined in Figure 3.

For a fixed randomly sampled protocol transcript $\delta \leftarrow \mathbb{S} \Delta$ and any probabilistic polynomial time adversaries \mathcal{A}_X and \mathcal{A}_Y

$$\text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \iff \text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \quad (7)$$

PFairnessGame₁(δ, λ) <hr/> 1 : $d'_Q \leftarrow \mathcal{A}_P(\delta)$ 2 : if $d_Q = d'_Q$ then return 1 3 : else return 0
--

Fig. 3: For an FE protocol between arbitrarily labelled parties P and Q , with a set of possible complete or partial protocol transcripts Δ , transcript $\delta \in \Delta$, and adversary acting as party P given by \mathcal{A}_P . If the adversary is able to extract the data d_Q from δ , they win.

Computational Strong Fairness requires that for a fixed transcript $\delta \leftarrow \mathbb{S} \Delta$, it is computationally intractable for X to extract d_Y if and only if it is computationally intractable for Y to extract d_X .

4 Two Party Fair Exchange

The key insight resulting in the protocol presented in this paper is that the impossibility of two party FE presupposes perfect security of *Strong Fairness* and a deterministic FE algorithm. By mirroring work in consensus[15], where robust protocols can be constructed by slightly relaxing the computational model, we have been able to construct a two party FE protocol with a strong notion of soundness.

The intuition behind the two party FE scheme presented in this paper is that we can produce a protocol in which there is a single synchronised success state, in which both parties learn about each other's secrets, and many failure states, where neither party learns anything. Participants run the protocol in sequential repeated rounds until they both arrive in a success state. If a malicious party disconnects from the protocol early, the cheated party can assume both participants have arrived in the synchronised success state and therefore infer some information to help derive the cheater's secret. After a short bruteforce attempt this guess can be validated, and so it is not vulnerable to genuine accidental disconnections.

4.1 Description

Party X , with secret d_X , and party Y , with secret d_Y , wish to exchange secrets. Both parties negotiate a pair of security parameters λ and λ' in advance, such

that λ is believed to be attackable in any reasonable time-frame for any reasonably equipped adversary and λ' should be attackable in a long but reasonable time frame for party X . This relies on Assumption 5, which states the computational capabilities of party X are reliably and publicly known to within an order of magnitude. For example, λ could have an expected time-to-bruteforce of 1000 years, but λ' is considered bruteforceable for X in one month. The expected time to attack $\mathcal{E}(\lambda)$ is denoted as T , and the expected time to attack $\mathcal{E}(\lambda')$ is denoted as T' . It should be the case that:

$$T' \ll T \quad (8)$$

As further setup, given λ and λ' both parties then derive value M , such that

$$M\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) = \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda') \quad (9)$$

Another way of expressing M is as the ratio between the time expected to attack the two instantiations of \mathcal{E} .

$$M = \frac{T}{T'} \quad (10)$$

With these parameters established, they can be reused as much as is required for as many domains as is relevant.

X and Y each generate a key each for a symmetric IND-CPA authenticated encryption scheme $\mathcal{E}(\lambda)$.

$$X : k_{X_0} \leftarrow \mathcal{E}.\text{KGen}(\lambda) \quad Y : k_Y \leftarrow \mathcal{E}.\text{KGen}(\lambda) \quad (11)$$

X then generates a second key, and Y samples a random ciphertext from C' , the range of function $\mathcal{E}(\lambda').\text{Enc}$.

$$X : k_{X_1} \leftarrow \mathcal{E}.\text{KGen}(\lambda) \quad Y : c_r \leftarrow \$ C' \quad (12)$$

Both parties then encrypt their respective secrets, X uses the first key they generated.

$$X : c_X \leftarrow \mathcal{E}.\text{Enc}_{k_{X_0}}(d_X) \quad Y : c_Y \leftarrow \mathcal{E}.\text{Enc}_{k_{Y_0}}(d_Y) \quad (13)$$

Both parties freely exchange the generated ciphertexts. Party Y then generates a key for encryption scheme $\mathcal{E}(\lambda')$, b . They use this value to encrypt their key, k_Y .

$$Y : b \leftarrow \mathcal{E}.\text{KGen}(\lambda') \quad (14)$$

$$Y : c_{k_Y} \leftarrow \mathcal{E}.\text{Enc}_b(k_Y) \quad (15)$$

Both parties randomly sample m such that $0 \leq m < M$ and construct large M -sized vector \mathbf{v} . X inserts k_{X_1} at every position in \mathbf{v} except at position m_X , where they insert k_{X_0} . Y inserts c_r is at every position in \mathbf{v} except at position m_Y , where they insert c_{k_Y}

$$X : m_X \leftarrow \$ \{0, \dots, M - 1\} \quad Y : m_Y \leftarrow \$ \{0, \dots, M - 1\} \quad (16)$$

$$X : \mathbf{v}_X \leftarrow [k_{X_1}; M] \quad Y : \mathbf{v}_Y \leftarrow [c_r; M] \quad (17)$$

$$X : \mathbf{v}_X[m_X] \leftarrow k_{X_0} \quad Y : \mathbf{v}_Y[m_Y] \leftarrow c_{k_Y} \quad (18)$$

X then uses oblivious transfer to learn the element at position m_X in Y 's vector \mathbf{v}_Y .

$$X : c'_{k_Y} \xleftarrow{OT} \mathbf{v}_Y[m_X] \quad (19)$$

Because of the computational indistinguishability of ciphertexts produced by $\mathcal{E}(\lambda')$ X is unable to mount an attack with a better-than-negligible-in- λ probability of success against c'_{k_Y} without more information. In order to learn d_Y X must continue the protocol. This is a slightly counter-intuitive notion and a more rigorous security argument is provided later in this paper.

Next, Y uses oblivious transfer to learn the element at position m_Y in X 's vector, \mathbf{v}_X .

$$Y : k_X \xleftarrow{OT} \mathbf{v}_X[m_Y] \quad (20)$$

Y can use k_X to attempt to decrypt c_X .

$$Y : d_X \text{ or } \perp \leftarrow \mathcal{E}.\text{Dec}(c_X) \quad (21)$$

If Y succeeds they can release b to X in good faith. X then uses b to decrypt the key k_Y , which they use to decrypt the secret d_Y .

$$X : k_Y \leftarrow \mathcal{E}.\text{Dec}_b(c'_{k_Y}) \quad (22)$$

$$X : d_Y \leftarrow \mathcal{E}.\text{Dec}_{k_Y}(c_Y) \quad (23)$$

If Y attempts to cheat they confirm for X that they have a bruteforceable ciphertext, $c'_{k_Y} = c_{k_Y}$, containing the key to decrypt Y 's secret. With this information X can justify mounting a bruteforce attack in reasonable time-frame T' .

If $m_X \neq m_Y$ neither party has learned any information relevant to decrypting the secret of the other party, and so the round failed and both parties need to try again, restarting from key generation. Before restarting Y must prove the failure to X by revealing their value $k_X = k_{X1}$, which they could have only obtained if $m_X \neq m_Y$.

A single round from the protocol is shown diagrammatically in Figure 4.

4.2 Proofs

Theorem 1 (Computational Completeness). *For honest X and Y the probability of successfully swapping their secrets is*

$$\Pr[X(d_Y) \wedge Y(d_X)] \approx 1 \quad (24)$$

Proof. On a single run of the FE protocol, X and Y successfully exchange data if and only if $m_X = m_Y$. These values are the index in the vector \mathbf{v} at which X and Y store information relevant to their secret, and it is also the element they access via oblivious transfer from the other party's vector \mathbf{v} .

As m is an integer in the range $0 \leq m < M$, the probability of exchanging data is on a single run of the protocol is:

$$\Pr[X(d_Y) \wedge Y(d_X)] = \frac{1}{M} \quad (25)$$

One round from the Two Party Fair Exchange Protocol

$X(\mathcal{E}, \lambda, \lambda', d_X, M)$	$Y(\mathcal{E}, \lambda, \lambda', d_Y, M)$
$\mathsf{k}_{X_0} \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda)$	$\mathsf{k}_Y \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda)$
$\mathsf{k}_{X_1} \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda)$	$c_r \leftarrow \$ C'$
$c_X \leftarrow \mathcal{E}.\mathsf{Enc}_{\mathsf{k}_{X_0}}(d_X)$	$c_Y \leftarrow \mathcal{E}.\mathsf{Enc}_{\mathsf{k}_Y}(d_Y)$
	c_Y
	c_X
	$b \leftarrow \mathcal{E}.\mathsf{KGen}(\lambda')$
	$c_{\mathsf{k}_Y} \leftarrow \mathcal{E}.\mathsf{Enc}_b(\mathsf{k}_Y)$
$m_X \leftarrow \$ \{0, \dots, M - 1\}$	$m_Y \leftarrow \$ \{0, \dots, M - 1\}$
$\mathbf{v}_X \leftarrow [\mathsf{k}_{X_1}; M]$	$\mathbf{v}_Y \leftarrow [c_r; M]$
$\mathbf{v}_X[m_X] \leftarrow \mathsf{k}_{X_0}$	$\mathbf{v}_Y[m_Y] \leftarrow c_{\mathsf{k}_Y}$
	$c'_{\mathsf{k}_Y} \xleftarrow{OT} \mathbf{v}_Y[m_X]$
	$\mathsf{k}_X \xleftarrow{OT} \mathbf{v}_X[m_Y]$
	$o \leftarrow \mathcal{E}.\mathsf{Dec}_{\mathsf{k}_X}(c_X)$
	if $o = \perp$ then $r \leftarrow \mathsf{k}_X$
	else $r \leftarrow b$
	r
if $r = \mathsf{k}_{X_1}$ then return \perp	if $o = \perp$ then return \perp
else $\mathsf{k}_Y \leftarrow \mathcal{E}.\mathsf{Dec}_r(c'_{\mathsf{k}_Y})$	else return o
return $\mathcal{E}.\mathsf{Dec}_{\mathsf{k}_Y}(c_Y)$	

Fig. 4: Overview of a single round of the two party fair exchange protocol.

By Assumption 4, M is a constant. It follows that the protocol can be repeated across as many rounds as is required for the probability of a successful data exchange to be arbitrarily close to 1.

$$\therefore \Pr[X(d_Y) \wedge Y(d_X)] \approx 1 \quad (26)$$

The following Lemmas prove qualities about the soundness of the system after each message is sent. In each subsequent Lemma the protocol transcript δ is extended with the next message as specified in the protocol description, and the adversary \mathcal{A} takes on the role of either participant X or Y , whichever is the recipient of the latest message. The first two lemmas are trivial, and reduce to an attack on the underlying encryption scheme used to construct the FE protocol.

$$\text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (27)$$

Proof. Consider adversary \mathcal{A}_X capable of extracting d_Y from δ . This could easily be used to construct \mathcal{B} , in Figure 5, which forwards ciphertexts to \mathcal{A}_X to provide a decryption.

$$\therefore \text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (28)$$

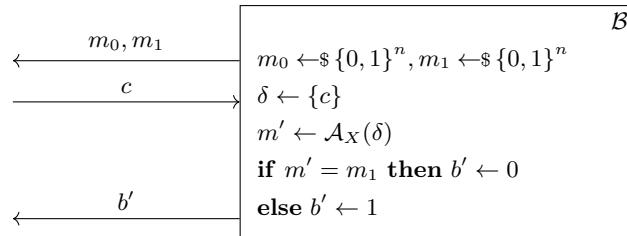


Fig. 5: Adversary \mathcal{B} ; capable of winning the IND-CPA security game given the existence of adversary \mathcal{A}_X .

Lemma 1. When $\delta = \{c_Y, c_X\}$

$$\text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (29)$$

This trivial proof is omitted for brevity. It follows the same outline as the proof for Lemma ?? but with party Y and X swapped, and \mathcal{B} randomly samples a placeholder c_Y in the construction of the transcript δ .

Lemma 2. When $\delta = \{c_Y, c_X, c_z\}$

$$\text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) = \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (30)$$

Proof. There are two possible situations; the first one is that $c_z = c_{k_Y}$, an encrypted version of the key used to produce ciphertext c_Y , the other is that they are completely unrelated and c_z is randomly sampled from C' . In the latter case this proof is trivial as the adversary has no more information than they did in Lemma ???. The former case occurs with probability $\frac{1}{M}$.

The definition of the constant M in the protocol is given by Equation 1:

$$\begin{aligned} M\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) &= \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \\ \text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) &= \frac{1}{M}\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \end{aligned} \quad (31)$$

plus some omitted negligible probability of attacking $\mathcal{E}(\lambda)$ as given in Lemma ??.

The definition of the constant M in the protocol is given by

$$M\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) = \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \quad (32)$$

It follows that the advantage of \mathcal{A}_X in the FairnessGame is

$$\therefore \text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) = \frac{1}{M}\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \quad (33)$$

$$= \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (34)$$

Lemma 2 assumes that the adversary only has access to a single partial transcript of the FE exchange and so it does not cover the possibility that the adversary gathers many samples and then attempts to identify and attack the weaker encryption of \mathcal{E} with a security parameter λ' . This attack has a negligible probability of success if \mathcal{E} is IND-CPA secure. This argument is given in more detail in Subsection 4.3.

Lemma 3. *When $\delta = \{c_Y, c_X, c_z, k_X\}$ and it is computationally intractable for party X to extract d_Y*

$$\text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (35)$$

Proof. As given c_z it is computationally intractable for X to extract d_Y , it must be the case that $c_z \neq c_{k_Y}$ and therefore $m_X \neq m_Y$. Because of this, k_X is completely unrelated to c_X and so is no help for decrypting it. Because of this, the adversary has no more information than they did in Lemma 1 and their advantage in the fairness game must be the same.

$$\therefore \text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (36)$$

Theorem 2 (Computational Strong Fairness). *Given \mathcal{E} , an IND-CPA secure authenticated encryption scheme, where $\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \leq \text{negl}(\lambda)$, in the two party FE protocol, if the ability of Y to extract d_X from transcript δ is computationally intractable, then for any probabilistic polynomial time adversary \mathcal{A}*

$$\text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \quad (37)$$

Similarly, if the ability of X to extract d_Y from transcript δ is computationally intractable, then for any probabilistic polynomial time adversary \mathcal{A} :

$$\text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \quad (38)$$

Proof. In the first case, when party Y is acting maliciously, the transcript must be either $\delta = \{c_Y, c_X\}$ or $\delta = \{c_Y, c_X, c_z, k_X\}$. The former case is proved in Lemma 1 and the latter is proved in Lemma 3. It follows that for all possible situations, if the ability of X to extract d_Y from δ is intractable then

$$\text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (39)$$

$$\implies \text{Adv}_{\mathcal{A}_Y}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \quad (40)$$

With a similar argument, when party X is acting maliciously Lemmas ?? and 2 prove the advantage of \mathcal{A}_X must be

$$\text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda) \quad (41)$$

$$\implies \text{Adv}_{\mathcal{A}_X}^{\text{fairness}}(\lambda) \leq \text{negl}(\lambda) \quad (42)$$

4.3 Sybil Attack Resistance

This second half of Theorem 2 seems counter-intuitive and warrants some further discussion. The theorem presupposes that an adversary only has access to a single transcript. It is not immediately obvious why X cannot mount a Sybil-style attack to gain many transcripts, then identify the ciphertext containing Y 's encryption key and mount a relatively simple bruteforce attack to extract the key. This attack would see X engaging in the protocol and disconnecting after the transcript is $\delta = \{c_Y, c_X, c_z\}$, and by repeating this process an expected M times, one of these samples should include $c_z = c_{k_Y}$.

Unfortunately for X they must compete with the fact that as \mathcal{E} is an IND-CPA secure encryption scheme the ciphertext produced by encrypting k_Y is computationally indistinguishable from a randomly sampled ciphertext from C' . As X cannot distinguish between c_{k_Y} and c_r randomly sampled from C' they must attempt the bruteforce attack against each sample in turn. Given that $\mathcal{E}(\lambda')$ is M times easier to attack than $\mathcal{E}(\lambda)$, but X is expected to perform M bruteforce attacks to extract d_Y a Sybil-style attack is clearly not feasible as it would take as much effort as a direct attack against $\mathcal{E}(\lambda)$.

It is therefore of vital importance that both parties regenerate cryptographic keys at the start of each repeat execution, and do not reuse them. Otherwise this attack vector would be trivial, as X would be able to correlate the most-significant bits of the key and identify the correct one as an outlier and attempt the easier bruteforce attack. This concept is made more concrete in Appendix A, where a theorem is provided proving the *Computational Strong Fairness* of the FE scheme under a Sybil-style attack given that ciphertexts produced by $\mathcal{E}(\lambda')$ are indistinguishable.

4.4 Evaluation

We have proved that the two party FE scheme has both the *Computational Completeness* and *Computational Strong Fairness* properties without the need for a trusted third party arbitrator.

A glaring deficiency in our scheme is that for a fixed λ' , as λ is made arbitrarily large, M must grow exponentially to reflect the difference between λ and λ' . With this, the probability of a successful secret exchange per round of the protocol becomes vanishingly small. λ , λ' , and M must be carefully chosen to balance the security of λ , the ease of the recovering from a cheater given by λ' , and the failure rate $\frac{M-1}{M}$, for a given setting. The massive failure rate of our scheme is mitigated by the fact that for a given setting M is a constant, and so the protocol is expected to be run a constant M times before a success.

5 Related Work

Early FE protocols relied on the technique of gradual disclosure, in which individual bits of a secret are released over many rounds[4]. In other works mutual disclosure has been extended to include the single-bit case, by simulating a biased coin which slowly reveals information about a secret over many rounds. This approach has the unfortunate side affect that early termination may leave one party, by sheer bad luck, with a misrepresentative sample of coin flips. This was later refined to simulate the production of a shared biased coin which when flipped would land on “same” or “different”, thereby ensuring both participants had an identical sample of flips[16]. By definition, no gradual disclosure schemes are able to satisfy any version of the *Strong Fairness* property.

Another early solution to the FE problem was constructed based on the assumption that Y is able to witness the behaviour of X after seeing d_Y , and is therefore able to discern d_X based on the course of action X took[11]. Interestingly, this assumption is a less realistic ability than one required in this work, wherein party X is able to witness a party Y disconnecting after successfully extracting d_Y . Importantly, in this work an accidental disconnection can be easily validated through the short bruteforce attack or with a proof presented at a later date, after the concerned parties reconnect.

There is a body of work solving the FE problem with financial incentives to encourage honesty[13]. In such work a misbehaving party must forfeit previously posted financial assets. The advent of blockchain has allowed a decentralised ledger to stand in for a trusted third party[10,12,7,8]. Etherium smart contracts can encode the behaviour of a trusted third party, and release Etherium tokens should certain conditions be met[6]. This is unfortunately very expensive to implement, relies on the security of a blockchain and the honesty of it’s nodes, and is restricted to the exchange of cryptocurrency tokens.

All previous FE protocols with any notion of *Strong Fairness* use a trusted third party in some way. The previous state-of-the-art was *Optimistic* as it only required a third party to mediate in the case of a dispute[14].

6 Conclusion

To the best of our knowledge this is the first FE protocol satisfying a notion *Strong Fairness* without requiring any third party involvement. This result was previously widely assumed to be impossible, but can in fact be made possible by slightly relaxing the model of computation. The widely cited impossibility result showed that no asynchronous deterministic two party FE protocol with the *Strong Fairness* property exists[18]. We demonstrate here that a non-deterministic two party FE protocol with *Computational Strong Fairness* can exist.

Our protocol requires a constant, but very large, repeated number of rounds to achieve the successful exchange of secrets. Therefore, it may never be considered practical to implement.

7 Further Work

The value of two-party FE with *Strong Fairness* is clear, which for the first time our construction shows is possible. However, as highlighted in the Evaluation, the protocol presented in this work is not without flaws. The massive per-round failure rate of $\frac{M-1}{M}$ requires the protocol to be run an expected M times. Depending on how cautiously M is selected and the security parameter of the “secure” encryption scheme λ , this can be very large. The security of the scheme is relegated to the realm of “good enough” in order to keep M relatively small. There is potential for further work in hardening the scheme in this regard.

The ability to resist a Sybil-style attack is contingent on an adversary being unable to differentiate ciphertexts produced under security parameter λ' . However, there is no reason this bound can not be further secured by sampling c_r from the range of $\mathcal{E}(\lambda).\text{Enc}$ and projecting c_{k_Y} into this larger space. This would also require that projections onto the range of $\mathcal{E}(\lambda).\text{Enc}$ are computationally indistinguishable from randomly sampled elements.

A Proof of Sybil Attack Resistance

This second half of Theorem 2 seems counter-intuitive and warrants some further discussion, it is not immediately obvious why X cannot mount a bruteforce Sybil-style attack[?]. This attack would see X engaging in the protocol and disconnecting after the transcript has three entries, $\delta = \{c_Y, c_X, k'_Y\}$. By repeating this process M times possibly with M false personas, as shown in Figure 6, it would be expected that one of these samples includes a k'_Y related to the key used by Y to encrypt c_Y .

Unfortunately for X if they cannot distinguish between keys used by Y those not used by Y they must attempt the bruteforce attack against each sample. Given that $\mathcal{E}(\lambda')$ is M times easier to attack than $\mathcal{E}(\lambda)$, but X is expected to perform M bruteforce attacks to extract d_Y , a Sybil-style attack is clearly not feasible as it would take as much effort as a direct attack against $\mathcal{E}(\lambda)$.

Sample 1:	$\{c_{Y1}, c_{X1}, c'_{Y1}\}$
Sample 2:	$\{c_{Y2}, c_{X2}, c'_{Y2}\}$
	\vdots
Sample 41:	$\{c_{Y41}, c_{X41}, c'_{Y41}\}$
Sample 42:	$\{c_{Y42}, c_{X42}, \mathbf{c}_{Y0}\}$
Sample 43:	$\{c_{Y43}, c_{X43}, c'_{Y43}\}$
	\vdots
Sample M :	$\{c_{YM}, c_{XM}, c'_{YM}\}$

Fig. 6: M samples gathered as part of a Sybil-style attack against this fair exchange scheme. In this case Sample 42 contains a weakly encrypted ciphertext containing the key Y used to encrypt their secret.

It is therefore of vital importance that both parties regenerate cryptographic keys at the start of each repeat execution, and do not reuse them. Otherwise X could correlate the most-significant bits of the key and identify the correct one as an outlier and attempt an easier bruteforce attack.

Definition 6 (Repeat Fairness). *For party X define the advantage in the RepeatXFairnessGame defined in Figure 7 as*

$$\text{Adv}_{\mathcal{A}_X}^{\text{repeatfairness}}(\lambda, \lambda') = 2 \Pr[\text{RepeatXFairnessGame}(\lambda, \lambda') = 1] - 1 \quad (43)$$

The following lemma proves that the ability to mount a Sybil-style attack against our FE scheme reduces to the ability to distinguish between ciphertexts produced under security parameter λ' .

Theorem 3 (Sybil Attack Resistance Upper Bound). *For any probabilistic polynomial adversaries \mathcal{A}_X and \mathcal{B}*

$$\text{Adv}_{\mathcal{A}_X}^{\text{repeatfairness}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \quad (44)$$

This proof is given by a reduction from the difficulty of the RepeatXFairnessGame in Figure 7 to the ability to win the IND-CPA Game in Figure 2.

Proof. Consider RepeatXFairnessGame, and the hypothetical adversary \mathcal{A}_X , given in Figure 8 capable of winning the game. \mathcal{A}_X can be used as a subroutine to construct \mathcal{B} , given in Figure 9, an efficient adversary in the IND-CPA Game.

\mathcal{B} operates by guessing $b = 1$, meaning that c is an encryption of the key m_0 under $\mathcal{E}(\lambda')$. \mathcal{B} , operating under this guess, then simulates FE transcripts

RepeatXFairnessGame₁(λ, λ')
<pre> 1 : $d_Y \leftarrow \\$ D$ 2 : $d'_Y \leftarrow \mathcal{A}_X()$^{OTranscript($d_Y$)} 3 : if $d_Y = d'_Y$ then return 1 4 : else return 0 </pre>

Fig. 7: The PFairnessGame from Figure 3 adapted to allow the adversary acting as X access to an oracle, $OTranscript(d_Y)$, which provides FE protocol transcripts of length 3 (i.e. $\delta = \{c_Y, c_X, c_z\}$). d_Y is initially sampled from the set of all possible input data D , and is then used to instantiate the oracle. Note that the abstract parties P and Q in the original game have been replaced with their concrete counterparts, X and Y , as this game directly related to the FE scheme presented in this chapter.

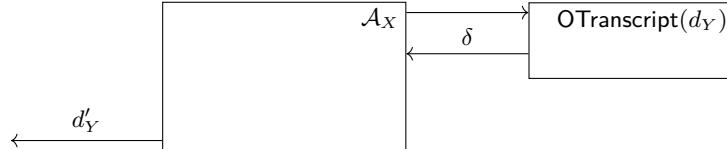


Fig. 8: Hypothetical adversary \mathcal{A}_X for the RepeatXFairnessGame, capable of winning with an expected M queries to $OTranscript$.

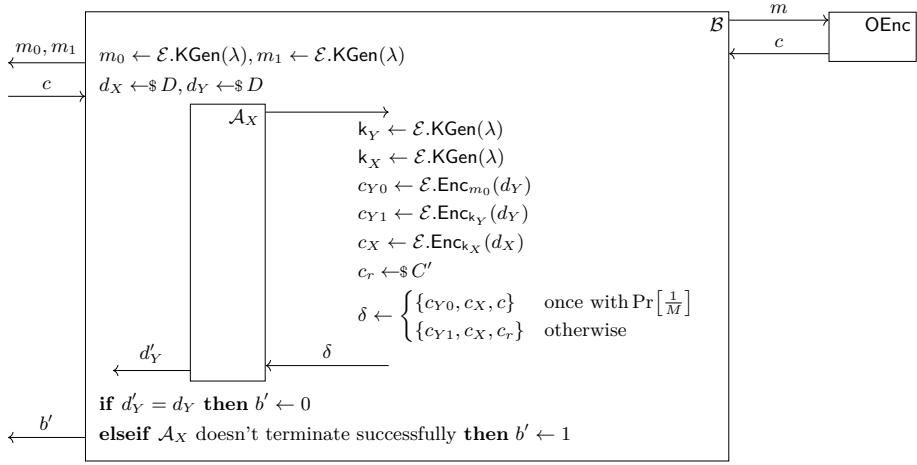


Fig. 9: Adversary \mathcal{B} ; capable of winning the IND-CPA security game given the existence of adversary \mathcal{A}_X . \mathcal{B} starts with the guess that $b = 0$, that the ciphertext c is an encryption of key m_0 . \mathcal{B} runs adversary \mathcal{A}_X as a sub-process, and simulates its oracle, OTranscript . The case in our FE protocol where $m_X = m_Y$ is simulated by providing $\delta = \{c_{Y0}, c_X, c\}$ where c_{Y0} is encrypted under m_0 and if \mathcal{B} 's guess is correct, c is an encryption of m_0 , this is provided to \mathcal{A}_X with probability $\frac{1}{M}$. Otherwise a transcript simulating $m_X \neq m_Y$ is provided with a randomly sampled c_r . If after $\mathcal{O}(M)$ calls to OTranscript \mathcal{A}_X has not terminated with a correct guess of d_Y there is a good chance \mathcal{B} 's guess that c is an encryption of m_0 is wrong. As M is constant, \mathcal{B} can run as many sub-processes of \mathcal{A}_X as they wish to improve their advantage to be arbitrarily close to 1.

for \mathcal{A}_X , which runs as a sub-process. This simulation operates by providing a transcript $\delta = \{c_{Y0}, c_X, c\}$ where c_{Y0} is encrypted under m_0 with probability $\frac{1}{M}$, and $\delta = \{c_{Y1}, c_X, c_r\}$ for randomly generated c_{Y1} and c_r with probability $\frac{M-1}{M}$. Note that the transcript containing c should only ever be sent once. If after $\mathcal{O}(M)$ calls to OTranscript \mathcal{A}_X has not successfully extracted secret $d_Y = d'_Y$ then there is a good chance that \mathcal{B} 's guess $b = 0$ is incorrect. As M is constant \mathcal{B} can repeatedly run \mathcal{A}_X a large number of times to gain an arbitrary degree of confidence in the correctness of their answer, b' .

$$\therefore \text{Adv}_{\mathcal{A}_X}^{\text{repeatfairness}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') \quad (45)$$

The upper bound on adversarial advantage provided by Lemma 3 is relatively unsatisfying. It would imply that the security of the two party FE scheme is no better than the weakest of the cryptosystems used, $\mathcal{E}(\lambda')$. The following arguments restrict this bound to prove stronger security notions are satisfied.

Consider a security game in which an adversary is provided with a set of n FE transcript samples of length three, i.e. transcripts of form $\delta = \{c_Y, c_X, c_z\}$, and must select which of the samples contains a c_z where c_z is the encryption of the key used to encrypt c_Y . This game can be further simplified by modifying it such that the adversary outputs two plaintext messages m_0 and m_1 and is then supplied with an ordered set S of size n containing $n - 1$ encryptions of m_0 and a single randomly placed encryption of m_1 . The adversary must identify the index of the encryption of m_1 . This security game is called NINDCPAGame and is defined more formally in Figure 10.

NINDCPAGame ₁ (λ, n)	
1 :	$S \leftarrow \emptyset$
2 :	$b \leftarrow \$ \{0, \dots, n - 1\}$
3 :	$k \leftarrow \mathcal{E}.\text{KGen}(\lambda)$
4 :	$m_0, m_1 \leftarrow \mathcal{A}()$
5 :	for i in $\{0, \dots, n - 1\}$
6 :	$S[i] \leftarrow \mathcal{E}.\text{Enc}_k(m_0)$
7 :	endfor
8 :	$S[b] \leftarrow \mathcal{E}.\text{Enc}_k(m_1)$
9 :	$b' \leftarrow \mathcal{A}(m_0, m_1, S)$
10 :	if $b = b'$ then return 1
11 :	else return 0

Fig. 10: The N-IND-CPA security game, in which an adversary must correctly identify the element in S which corresponds to an encrypted m_1 , where all the other elements are an encrypted m_0 .

Definition 7 (N-IND-CPA). *The advantage of a probabilistic polynomial time adversary in the NINDCPAGame security game, as defined in Figure 10, is defined as:*

$$\text{Adv}_{\mathcal{B}}^{\text{nind-cpa}}(\lambda, n) = n \Pr[\text{NINDCPAGame}(\lambda, n) = 1] - 1 \quad (46)$$

Intuitively, when $n \leq M$ it is expected that $\text{NINDCPAGame}(\lambda', n)$ is an easier game for the adversary than $\text{RepeatXFairnessGame}(\lambda, \lambda')^n$ as defined in Figure 7 with the additional caveat of n -bounded calls to OTranscript . This intuition is rooted in the idea that identifying a lightly encrypted version of the key, c_{k_Y} , is surely easier than extracting the strongly encrypted data encrypted from c_Y . The $\text{RepeatXFairnessGame}$ with calls to the transcript oracle bound by n is denoted $\text{RepeatXFairnessGame}^n$, and the adversarial advantage is denoted by $\text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda)$. The formal proof of the reduction from NINDCPAGame to $\text{RepeatXFairnessGame}^n$ follows.

Lemma 4. *Assuming that $\mathcal{E}(\lambda)$ is too strong to be directly attacked, for any n in the range $2 \leq n \leq M$ the advantage of a probability polynomial time adversary in the $\text{RepeatXFairnessGame}^n$ is less than advantage of a probabilistic polynomial time adversary in the NINDCPAGame with security parameter λ .*

$$\text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{nind-cpa}}(\lambda', n) \quad (47)$$

Proof. This can be easily shown by a reduction from the n -bounded $\text{RepeatXFairnessGame}$ to the NINDCPAGame. Consider an adversary \mathcal{A} capable of winning the n -bounded $\text{RepeatXFairnessGame}$ with non-negligible probability. This interface for this adversary has already been described previously in Figure 8 as part of an earlier proof, with the difference that it will make no more than n queries to OTranscript .

Adversary \mathcal{A} can be used to construct adversary \mathcal{B} as given in Figure 11 which is able to win the NINDCPAGame. As attacking $\mathcal{E}(\lambda)$ is considered infeasible, \mathcal{A} is unable to extract a value for d_Y until it sees a transcript of the form $\delta = \{c_Y, c_X, c_z\}$ where $c_Y = \mathcal{E}(\lambda).Enc_{m_1}(i)$ and $c_z = \mathcal{E}(\lambda').Enc(m_1)$ at which point it is able to extract the plaintext from c_Y . The equivalence between these two problems requires that n is upper bounded by M , otherwise the distribution of transcripts being sent to \mathcal{A}_X deviates from the expected distribution for the $\text{RepeatXFairnessGame}^n$. The other transcript samples provided to \mathcal{A} feature valid transcripts which have uncorrelated values for c_Y and c_z . Therefore

$$\therefore 2 \leq n \leq M, \text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{nind-cpa}}(\lambda', n) \quad (48)$$

The next step in achieving a much tighter bound on the security for the two party fair exchange scheme involves making the reasonable assumption that there is no better adversarial algorithm to the NINDCPAGame then applying an adversary \mathcal{A} against the IND-CPA security game to each element in S in turn until an encryption of m_1 is found. This assumption seems reasonable as checking set ownership of an element for random set S is $\mathcal{O}(|S|)$.

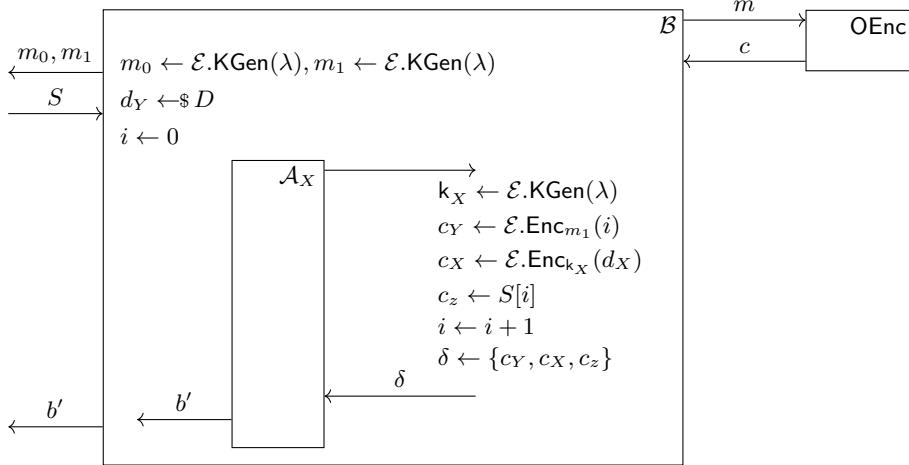


Fig. 11: An adversary \mathcal{B} capable of winning the NINDCPAGame given an adversary \mathcal{A} capable of winning the n -bounded RepeatXFairnessGame. Whenever \mathcal{A} requests a sample transcript the c_Y component is set to an encryption of the index i . When \mathcal{A} is able to correlate the key used to encrypt the data they extract and return the index.

Assumption 6 (Attacking NINDCPAGame) *There is no better algorithm to attack the NINDCPAGame than applying an adversary subroutine against the IND-CPA security game to each successive element of set S in turn until an encryption of m_1 is found.*

This assumption implies an equivalence in the adversarial advantage against the NINDCPAGame in λ' and the IND-CPA security game in λ . In order to prove this equivalence first an intermediate proof is required, which relates NINDCPAGame in λ' to the probability of a sequence of successes in successive the IND-CPA security games in λ' .

Lemma 5. *Given Assumption 6, the advantage of an adversary in the NINDCPAGame is given by the probability of an adversary winning $\frac{n}{2}$ successive IND-CPA security games.*

$$\text{Adv}_{\mathcal{A}}^{\text{nind-cpa}}(\lambda', n) = (\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \quad (49)$$

Proof. Assumption 6 states there is no better algorithm for attacking the NINDCPAGame than applying an IND-CPA adversary, \mathcal{B} , to each element of the challenge set S in turn until an encryption of m_1 is found. This requires not only that the adversary is able to correctly identify the single correct encryption of m_1 but also that they get no false positives on the path to the correct answer. Given this, the \mathcal{B} sub-processes must run successfully an average of $\frac{|S|}{2}$ times before

a valid encryption of m_1 is found. As $n = |S|$ the expected advantage of an adversary \mathcal{A} in the NINDCPAGame is

$$\therefore \text{Adv}_{\mathcal{A}}^{\text{nind-cpa}}(\lambda', n) = (\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \quad (50)$$

With the intermediate lemma, Lemma 5, it becomes easier to relate NINDCPAGame in λ' and the IND-CPA security game in λ . This reduction requires constraining M in order to prove valid values for n exist while satisfying the bounds introduced in Lemma 4.

Lemma 6. *Given that $1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)$, there exists some $n_0 \leq M$ such that for all $n > n_0$*

$$(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (51)$$

Proof. By Assumption 4

$$\frac{1}{M} \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda') = \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (52)$$

Considering the values for n for which

$$(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (53)$$

$$\leq \frac{1}{M} \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda') \quad (54)$$

involves solving the following equation for n in terms of M , thus finding a value for n_0 . As a probability raised to an arbitrarily large power is always smaller than the same probability divided by a fixed fraction for a power larger than some minimum value, a numerical value for n_0 must exist. The value of n_0 can be calculated given $\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')$ and M .

$$(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n_0}{2}} = \frac{1}{M} \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda') \quad (55)$$

$$\frac{n_0}{2} = \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(\frac{1}{M} \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')) \quad (56)$$

$$n_0 = 2 \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(\frac{1}{M} \text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')) \quad (57)$$

$$= 2(\log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(\frac{1}{M}) + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))) \quad (58)$$

$$= 2(\log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(\frac{1}{M}) + 1) \quad (59)$$

$$= 2(\log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(1) - \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M) + 1) \quad (60)$$

$$= 2(1 - \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)) \quad (61)$$

Therefore, for all $n > n_0$, where $n_0 = 2(1 - \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M))$

$$\therefore \exists n_0 \forall n \geq n_0 (\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (62)$$

A bound for M can be generated such that $n_0 \leq M$ given the following sequence of equations.

$$n_0 \leq M \quad (63)$$

$$2(1 - \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)) \leq M \quad (64)$$

$$1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M) \quad (65)$$

Therefore given $1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)$ for every M there exists an $n_0 \leq M$ such that for all $n \geq n_0$

$$\therefore \forall M \text{ such that } 1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M), \exists n_0 \leq M, \forall n > n_0, \quad (66)$$

$$(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (67)$$

Finally, we can use this sequence of lemmas to provide a tighter bound on the security of the two party fair exchange scheme.

Theorem 4 (Repeat Fairness Tighter Bound). *While $1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)$ the advantage of a probabilistic polynomial time adversary in the RepeatXFairnessGameⁿ is no more than the advantage of a probabilistic polynomial time adversary in the IND-CPA security game with security parameter λ .*

$$\text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (68)$$

Proof. By Lemma 4

$$2 \leq n \leq M, \text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda', n) \quad (69)$$

By Lemma 5

$$\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda', n) = (\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \quad (70)$$

and so

$$2 \leq n \leq M, \text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq (\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \quad (71)$$

By Lemma 6

$$\forall M \text{ such that } 1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M), \exists n_0 \leq M, \forall n > n_0, \quad (72)$$

$$(\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda'))^{\frac{n}{2}} \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (73)$$

Therefore given that $1 \leq \frac{M}{2} + \log_{\text{Adv}_{\mathcal{A}}^{\text{ind-cpa}}(\lambda')}(M)$ applying the transitive property to Equation 71 and Equation 73 it must be the case that

$$\therefore \text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-n}}(\lambda, \lambda') \leq \text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda) \quad (74)$$

By bounding the number of calls the adversary can make to the transcript oracle by some value $n \leq M$, we can guarantee the security is no worse than that of the stronger cryptosystem used during the construction of the two party fair exchange scheme, $\mathcal{E}(\lambda)$.

An interesting byproduct of Theorem 4 is that the predicate from Lemma 4 “assuming that $\mathcal{E}(\lambda)$ is too strong to be directly attacked” can be discarded. Before, this predicate was required to ensure the adversary is required to attack $\mathcal{E}(\lambda')$ over $\mathcal{E}(\lambda)$ but due to the fact that for all n larger than $n_0 \text{Adv}_{\mathcal{A}}^{\text{repeat-fairness-}n}(\lambda, \lambda')$ is bounded by $\text{Adv}_{\mathcal{B}}^{\text{ind-cpa}}(\lambda)(\lambda)$ regardless, the predicate “assuming that $\mathcal{E}(\lambda)$ is too strong to be directly attacked” becomes meaningless.