Fair Exchange for Decentralized Autonomous Organizations via Threshold Adaptor Signatures

Ruben Baecker[1], Paul Gerhart[2], Jonathan Katz*[3], and Dominique Schröder[1,2]

[1] *Friedrich-Alexander-Universität Erlangen-Nürnberg*
[2] *TU Wien*
[3] *Google*

Abstract

A *Decentralized Autonomous Organization* (DAO) enables multiple parties to collectively manage digital assets in a blockchain setting. We focus on achieving *fair exchange* between DAOs using a cryptographic mechanism that operates with minimal blockchain assumptions and, crucially, does not rely on smart contracts.

Specifically, we consider a setting where a DAO consisting of n_{S} sellers holding shares of a witness w interacts with a DAO comprising n_{B} buyers holding shares of a signing key sk; the goal is for the sellers to exchange w for a signature under sk transferring a predetermined amount of funds. Fairness is required to hold both *between* DAOs (i.e., ensuring that each DAO receives its asset if and only if the other does) as well as *within* each DAO (i.e., ensuring that all members of a DAO receive their asset if and only if every other member does).

We formalize these fairness properties and present an efficient protocol for DAO-based fair exchange under standard cryptographic assumptions. Our protocol leverages *certified witness encryption* and *threshold adaptor signatures*, two primitives of independent interest that we introduce and show how to construct efficiently.

1 Introduction

A *Decentralized Autonomous Organization* (DAO) is a group of several parties who jointly exercise control over digital assets in a blockchain setting. Real-world examples such as MakerDAO and Uniswap's DAO oversee assets with substantial value, as evidenced by the price of their underlying governance tokens. For example, the governance token for Uniswap, UNI, has a market capitalization of roughly $5.7 billion according to CoinDesk. The governance of a DAO is distributed among its members, with pivotal decisions (such as the purchase, sale, or exchange of assets) being collectively determined by the members of the DAO through a voting process. The implementation of DAOs frequently employs smart contracts, thereby ensuring transparent adherence to predefined rules. In this work, we explore whether certain functions of a DAO can be achieved via cryptographic mechanisms *without relying on smart contracts*. Such an approach is essential for ensuring compatibility with blockchains such as Bitcoin and Monero that lack smart-contract functionality.

*Portions of this work were done while at the University of Maryland.

Fair exchange of digital assets. We focus on fair exchange of digital assets between DAOs. Specifically, we consider a setting where there is a *seller DAO* composed of n_{S} sellers who hold shares of a witness w for a public statement, and a *buyer DAO* consisting of n_{B} buyers who hold shares of a signing key sk controlling a digital asset. The members of the seller DAO interact with the members of the buyer DAO with the goal of exchanging w for a signature under sk authorizing the transfer of a predetermined amount of funds. After the exchange, all members of the buyer DAO should learn w, and all members of the seller DAO should receive an equal share of the transferred funds.

Security requirements. In the context of fair exchange between DAOs, there is an inherent lack of trust between the parties. Sellers are concerned that buyers may not fulfill their payment obligation, while buyers fear that sellers might fail to deliver the witness as promised. To ensure fairness, we require that the protocol guarantees fairness for the buyer DAO even if all sellers are corrupted and, likewise, fairness for the seller DAO even if all buyers are corrupted. Additionally, fairness must hold *within* each DAO: malicious buyers should not be able to learn the witness while preventing honest buyers from learning it, and similarly, malicious sellers should not be able to receive funds while blocking honest sellers from doing so. In all cases, we consider an adversary capable of corrupting both buyers and sellers. We formalize these fairness properties as two key security requirements, defined formally in Section 3.2.

Buyer fairness: An adversary who controls all the sellers should be unable to obtain the funds of any honest buyers unless those honest buyers successfully learn the witness w. This should hold even if the adversary additionally controls up to some threshold of buyers.

Seller fairness: Analogously, an adversary who controls all the buyers should be unable to learn any information about w unless all honest sellers receive their expected funds. This should hold even if the adversary additionally controls up to some threshold of sellers.

We additionally require the following, which is not implied by the above:

Witness privacy: An adversary who corrupts up to some threshold of sellers but no buyers[1] should not learn any information about the witness w (even if the sellers receive payment).

Witness privacy ensures that *only* the buyers learn w, and w is not revealed to sellers (or external eavesdroppers) in the course of the protocol.

1.1 Our Contributions

We summarize our contributions at a high level, and refer the reader to Section 2 for a more detailed technical overview.

Formalization. We introduce a formal framework for the problem of fair exchange in decentralized autonomous organizations (DAOs). While previous works have explored different formulations of fair exchange (cf. Section 1.2), our approach is the first to specifically address its applicability to DAOs.

[1]Every buyer participating in the protocol learns the witness, so we must assume the adversary does not corrupt any buyers here.

Building Blocks. We introduce two novel cryptographic building blocks:

- We introduce and construct *threshold adaptor signatures*[Erw+21; Aum+21].[2] Our security definitions align with the most recent formalizations of adaptor signatures [Ger+24a], which address gaps in earlier works.
- We put forward the notion of *certified witness encryption*, which allows for the encryption of a message based on a verification key/message pair (vk, m) such that decryption is only possible given a valid signature on m relative to vk. (This can be viewed as a form of witness encryption [Gar+13] for a specific NP language, thus allowing for an efficient realization.) This primitive is particularly useful for achieving fairness in a blockchain setting, where a signature on some transaction must be posted on a public blockchain in order to be processed; once posted, anyone holding the corresponding certified witness encryption of a message will be able to recover that message. We also provide a framework for constructing certified witness encryption for a large class of signature schemes based on the discrete-logarithm problem [Sch91; BBS04; CL03; BSW06; KW03].

Possibility of Fair Exchange. We introduce a *fair-exchange protocol* that utilizes *timelocks* and standard cryptographic assumptions, guaranteeing security against any level of buyer corruption. The protocol assumes a threshold t_{S}, requiring the participation of at least t_{S} honest sellers. Our protocol is highly efficient, operates without the need for smart contracts, and is built upon our newly introduced primitives: threshold adaptor signatures and certified witness encryption.

Imossibility of Fair Exchange. We complement the above with an impossibility result demonstrating that a fair-exchange protocol cannot achieve seller fairness in the presence of a minimal blockchain. In addition, in the presence of a minimal blockchain with timelocks, we show that a fair-exchange protocol cannot achieve seller fairness if not at least t_{S} honest sellers participate in the exchange. This result establishes that our assumption of an honest majority of participating sellers is optimal with respect to the minimal assumptions needed for blockchain functionality.

1.2 Related Work

There is an extensive literature on fair exchange in various settings that we cannot hope to survey here; instead, we focus on the most directly relevant prior work. The key feature distinguishing our work from most prior work on fair exchange is that rather than consider the exchange of items between two parties or among multiple parties, we consider the exchange of items between two *groups* of parties with a threshold requirement within each group. We stress, in particular, that this setting requires consideration of *intra*-group fairness and not just inter-group fairness as in prior work, and also requires consideration of an adversary who may corrupt some parties in *both* groups. The only prior work we are aware of in this setting assumes smart contracts [EFS20; DEF18].

[2] Concurrent and independent work by Ji et al.[Ji+24] and Kaijta et al.[KOT24] also explores threshold adaptor signatures. However, their definitions rely on prior formulations of adaptor signatures that have been shown to be unsuitable for practical applications[Ger+24a]. Additionally, the latter work is restricted to the n-out-of-n case.

Several prior works have considered fair exchange (though not our variant, as discussed above) in the presence of a blockchain. Secure computation with penalties [And+14; And+13; BK14; KMB15; BDD20] does not guarantee fairness but does ensure that parties who violate fairness are "penalized" by losing funds. Thus, that approach can achieve fairness against a rational adversary whose utilities are assumed to be known, but not necessarily against a malicious adversary as we consider here. Moreover, it is not clear how to extend that work to our setting, where it may be possible to tell, e.g., that *some* seller cheated but it may not be clear which seller to penalize (and it would be unfair to penalize them all). Other work [Cho+17; Gad+23] shows how to achieve fairness in the presence of a blockchain utilizing trusted hardware or strong cryptographic assumptions (witness encryption for all of NP).

Adaptor signatures [Poe17; Aum+21; Erw+21; DOY22; Ger+24b] have been shown to allow for the fair exchange of a witness for a signature between two parties in the presence of a blockchain. Our work can be viewed as extending that work to the case of two *groups* of parties with a threshold requirement. While it is natural to replace the adaptor signature with a threshold adaptor signature to do so, we stress that this is not sufficient to solve the fairness problem due to the added challenge of ensuring intra-group fairness of the buyers.

In the two-party setting—with the two parties corresponding to the buyer DAO and the seller DAO in our application—fairness [Cle86] (or even partial fairness [GK10]) is impossible to achieve via generic secure computation. Although various relaxations of fair MPC have been considered, they would all provide unacceptable security in our context.

1.3 Overview of the Paper

We give a high-level technical overview of our solution in Section 2. In Section 3, we formalize fair exchange in our setting. Before constructing our fair-exchange protocol, we introduce two essential building blocks: threshold adaptor signatures (Section 4) and certified witness encryption (Section 5). With these primitives established, we present our fair-exchange protocol in Section 6. Finally, we show our impossibility result in Section 7.

2 Technical Overview

As a starting point, observe that adaptor signatures [Poe17; Ger+24b] provide a solution for fair exchange between two DAOs that each contain a single party. Recall that an adaptor signature is defined relative to an NP relation $\mathcal{R}$ and a signature scheme Π_{sign}. Fair exchange can be achieved by having the buyer (who holds a signing key) create a pre-signature with respect to a particular NP statement, and then sending the pre-signature to the seller. The seller converts the pre-signature into a valid signature σ_{coin} using the corresponding witness w it knows, and posts the signature on the blockchain; at that point, the seller is paid. Upon observing σ_{coin} on the blockchain, the buyer is able to learn w. Our goal is to extend this approach to two independent DAOs, one for a set of sellers and one for set of buyers, such that each set has an independent threshold requirement for executing the exchange. We first generalize the set of buyers, followed by the set of sellers, and discuss the resulting technical challenges of our approach.

2.1 The Buyer DAO

To generalize to a set of buyers, we extend adaptor signatures to the threshold setting, while ensuring compatibility with existing blockchains such as Bitcoin by prioritizing natively supported solutions. However, solving our problem requires additional cryptographic tools beyond (threshold) adaptor signatures alone.

Threshold Adaptor Signatures We introduce a threshold variant of adaptor signatures that allows a given subset of signers to jointly compute a pre-signature. To maintain compatibility with the underlying blockchain, the resulting pre-signature must be indistinguishable from a single-user pre-signature. In addition, our thresholding scheme preserves the core functionality of classical adaptor signatures: (i) given a pre-signature $\widetilde{\sigma}$ and a witness w, one can derive a full signature σ (adaptation), and (ii) given a pre-signature $\widetilde{\sigma}$, a signature σ, and a statement stmt, one can extract the witness (extraction).

Our construction is based on the Schnorr threshold signature scheme Sparkle [CKM23]. In this scheme, each signer samples a random value $r_i \leftarrow\!\!\$\ \mathbb{Z}_p$, computes the partial nonce $R_i = g^{r_i}$, commits to it using the hash function H_{cm}, and shares the commitment with the other signers. Once all commitments have been received, the signers reveal their nonces and check for consistency. After collecting all nonces R_i, they compute partial pre-signatures by first verifying the commitments and aggregating the combined nonce as

$$R = \prod R_i.$$

We follow their basic strategy, but we multiply R with the statement s in the derivation of the Fiat-Shamir challenge h as:

$$h = \mathsf{H}_{\mathsf{Sign}}(\mathsf{pk}, m, R \cdot s).$$

Note that s must be included for security reasons. Each signer then computes a partial pre-signature

$$z_i = r_i + h \cdot \lambda_i \cdot \mathsf{sk}_i,$$

where λ_i is the Lagrange coefficient. Aggregating the partial presignatures yields the final presignature

$$z = \sum_i z_i = r + h \cdot \mathsf{sk},$$

where r is the combined randomness and sk is the aggregated signing key. The aggregated pre-signature has the form

$$\widetilde{\sigma} = (R \cdot s, z).$$

As with the single-signer Schnorr adaptor signature setting, this pre-signature can be adapted into a full signature using a valid witness w for the statement s. Given $\widetilde{\sigma}$ and w, the adaption is computed as

$$\sigma = (R \cdot s, z + w).$$

Since the nonce R is homomorphic, this adaptation implicitly results in a valid Schnorr signature corresponding to the nonce

$$R \cdot s = g^{r+w}.$$

Certified witness encryption (CWE). Threshold adaptor signatures ensure fairness between a set of honest buyers and a *single* seller, however, they do not inherently guarantee fairness among the buyers themselves. In particular, a malicious buyer could abort the pre-signing protocol after collecting the necessary partial pre-signatures from others without revealing her own pre-signature, which would allow the adversary to compute a pre-signature while leaving the remaining buyers unable to do so. The malicious buyer can forward this pre-signature to the (possibly malicious) seller, who can convert it into a valid signature σ_{coin} that it posts on the blockchain, thus receiving the payment. All honest buyers have now paid but cannot extract the witness w using σ_{coin} because they do not know the corresponding pre-signature $\widetilde{\sigma}$. We address this "intra-buyer" fairness problem by introducing a primitive called *certified witness encryption*. At a high level, a CWE scheme allows for encrypting a message relative to a signature verification key vk and a designated message m. The resulting ciphertext can be decrypted by any party who knows a valid signature on m with respect to vk. By using a CWE scheme, we can achieve fairness for all honest buyers in the following way. Before disclosing their partial pre-signatures, all buyers encrypt their partial pre-signatures using the CWE scheme and forward the resulting ciphertext alongside a proof of well-formedness to all other buyers. This ciphertext can be decrypted by anyone who knows σ_{coin}. After receiving and verifying enough CWE ciphertexts, each honest buyer forwards its partial pre-signature to the other buyers. Now, if the seller ever posts σ_{coin} to the blockchain to receive a payment, all the buyers will be able to decrypt the CWE ciphertexts they received to learn the corresponding partial pre-signatures; this allows them to reconstruct the pre-signature and then (using σ_{coin}) extract the desired witness.

We construct a CWE scheme for Schnorr signatures to ensure compatibility with our Schnorr threshold adaptor signature scheme. The key observation towards building a CWE scheme from Schnorr lies in the Schnorr verification equation. Given a Schnorr signature $\sigma = (R, z) = (R, \mathsf{sk} \cdot h + r)$, where $h = \mathsf{H}_{\mathsf{Sign}}(\mathsf{pk}, R, m)$, verification is performed by checking the equation

$$g^z = \mathsf{pk}^h \cdot R.$$

While $\mathsf{pk}^h \cdot R$ can be computed publicly using only the nonce R, public key pk, and message m, obtaining the signature effectively requires computing the discrete logarithm $z = \mathsf{sk} \cdot h + r$ which is effectively possible knowing the secret and the DLog r of R. We leverage this discrete logarithm relation to build encryption and decryption keys for our CWE scheme. To encrypt a message m_{Enc} that can be decrypted knowing a Schnorr signature valid for $(\mathsf{pk}, m_{\mathsf{Sign}}, R)$, a party computes the encryption key

$$\mathsf{ek} = \mathsf{pk}^h \cdot R$$

and constructs the ciphertext

$$ct = \langle g^y, \mathsf{ek}^y \cdot m_{\mathsf{Enc}} \rangle,$$

where $y \in \mathbb{Z}_p$ is a uniformly sampled random value. Effectively, this ciphertext is an ElGamal ciphertext w.r.t. the key ek. Given a valid signature (R, z) and the ciphertext ct, decryption follows standard ElGamal decryption using the decryption key $\mathsf{dk} = z$. While this construction is both efficient and simple, it satisfies the requirements of a CWE scheme and ensures intra-buyer fairness in a threshold pre-signing protocol.

Following this blueprint, we extend CWE to a broad class of signature schemes, including Camenisch-Lysyanskaya (CL) [CL03], Waters+ [BSW06], and Katz-Wang [KW03].

2.2 The Seller DAO

At first glance, the challenge of extending the setting from a single seller to multiple sellers seems symmetrical to the case of multiple buyers. Consequently, one might expect that similar techniques could be used to solve the problem. However, it turns out that the problem is fundamentally different. To illustrate this issue, consider a scenario where multiple sellers each possess a share of the witness w, and assume that the buyers have already provided a valid pre-signature $\widetilde{\sigma}$. A seemingly natural approach would be to execute a protocol among the sellers to reconstruct the witness w. Given the properties of threshold adaptor signatures, once the sellers obtain w and have a pre-signature $\widetilde{\sigma}$, they can efficiently compute the signature σ_{coin} and post it on the blockchain, which authorizes the payment to the sellers. However, fairness emerges as a critical concern — this time among the sellers.

Impossiblity of seller fairness with minimal blockchains. Unlike the single-seller case, in the multi-seller case, it is in the seller's interest not only to receive payment from the buyers but also to gain access to the full witness w *without* triggering payment from the buyers. In particular, an adversary who corrupts all buyers and a single seller can gain access to σ_{coin} before any other seller, thereby undermining the fairness of the exchange. Specifically, such an adversary could use this knowledge to reconstruct the witness and spend the funds before σ_{coin} is published to the blockchain. As a result, σ_{coin} becomes invalid, preventing the remaining sellers from receiving their payments—an attack we call frontrunning. Moreover, it is impossible to detect the dishonest seller in this setting, since they follow the protocol without any detectable deviation. However, their actions still allow the adversary to prematurely spend the buyer's DAO coin. This raises the fundamental question of whether fairness can be achieved in a setting with a minimal blockchain that lacks scripting support. Unfortunately, our first result in this direction is negative. As we demonstrate in Section 7, frontrunning cannot be prevented in a minimal blockchain.

(Imp)Possibility of seller fairness with blockchains that support time locks. Since achieving fairness for distributed sellers is impossible in the context of scriptless blockchains due to frontrunning, we incrementally improve functionality in non-trivial ways — acknowledging that simply enabling smart contracts would trivially solve the problem. Consequently, we consider blockchains that support time-lock puzzles to prevent frontrunning. A time-lock puzzle allows buyers to lock their coins in such a way that they can either be spent with a certain signature or automatically refunded after a certain period of time. Our goal is to mitigate frontrunning by ensuring that an adversary cannot spend the funds associated with σ_{coin}. Instead, the timeout is configured so that an honest seller who has received a sufficient number of shares is guaranteed to complete payment before the funds become available to an adversarial entity. This mechanism preserves fairness among sellers since no single corrupt seller can undermine the exchange by prematurely accessing or invalidating the buyer's DAO funds. Unfortunately, this intuitive approach does not work in general:

Impossibility in the presence of fewer than t_{S} honest sellers. If the protocol works with less than t_{S} honest sellers, fairness is not guaranteed. In this case, witness reconstruction fails, preventing the protocol from completing on the seller's side. Fairness for sellers requires that once the protocol starts and at least one honest share is exchanged, it must be completed to prevent the adversary from gaining an unfair advantage.

An adversary who controls up to $t_{\mathsf{S}} - 1$ sellers needs only one additional share from an

honest seller to reconstruct the witness w. Since sellers must eventually reveal their shares to publish σ_{coin}, the adversary can exploit this by identifying and corrupting the first recipient of a message in the protocol that allows this party to compute the missing share, thus obtaining w before any honest party. After obtaining the missing share, the adversary can withhold his own by not sending further messages in the protocol, preventing the remaining honest sellers (less than t_{S}) from reconstructing w. This blocks the completion of the protocol on the sellers' side. To ensure fairness, buyers reclaim their money if the protocol remains incomplete. Thus, the adversary can both prematurely extract w and disrupt completion, undermining fairness. As a result, the protocol ends in a state where the buyers did not pay the sellers, yet the dishonest sellers successfully learned the witness. This outcome fundamentally violates seller fairness.

Possibility in the case of t_{S} honest sellers. On the positive side, we show that seller fairness is achievable if at least t_{S} sellers participate in the fair exchange protocol. Specifically, we describe a protocol that ensures fairness for both sellers and buyers. The protocol begins with buyers locking their coins on the blockchain and generating a pre-signature—corresponding to the locked coins—using a threshold adaptor signature scheme. This pre-signature is then distributed to the sellers. Each seller subsequently shares its witness share with all other sellers. Assuming that at least t_{S} sellers participate, at least one honest seller will receive enough shares to reconstruct the witness before the lock expires. This seller can then use the witness to finalize the pre-signature, obtaining σ_{coin}, and publishing it to the blockchain to complete the payment. To ensure that sellers can successfully claim their payments, honest sellers participate in the protocol only after verifying that the coins are locked for a sufficient duration upon receiving the pre-signature. Moreover, they engage in the protocol only if they are certain that at least t_{S} honest sellers are involved. This can be achieved through a voting mechanism conducted *before* the fair exchange protocol begins. The voting effectively checks, if at least $2 \cdot t_{\mathsf{S}} - 1$ sellers vote for the execution. If the lock duration is insufficient or if the required quorum of t_{S} honest participating sellers is not met, they refrain from participating and instead wait for the lock to expire, thereby preventing any unfair exchange.

Witness privacy. In the approach described above, each seller learns the witness during protocol execution, and even passive eavesdroppers can gain access to it. This is highly undesirable as it compromises confidentiality and control over the witness. To mitigate this problem, we introduce a blinding mechanism. The buyers collaboratively compute a blinding value, which is then used to re-randomize the exchanged witness. Instead of directly revealing the original witness, the sellers exchange only a blinded version, ensuring that they do not gain any information about the true value of the witness. Once the exchange is complete, all buyers can locally remove the blinding factor to recover the original witness, preserving both security and fairness in the protocol.

In summary, our positive results show that both seller and buyer fairness can be achieved when the protocol is run on blockchains that support time locks and under the assumption of t_{S} honest participating sellers. Our impossibility results complement this by proving that both conditions — time locks and t_{S} honest participating sellers — are not only necessary but also sufficient for fairness to hold.

3 Defining Fair Exchange

Multiparty Fair Exchange (MPFE) protocols provide a cryptographic framework for decentralized autonomous organizations (DAOs) to trade assets. An MPFE protocol involves multiple sellers and buyers, with sellers attempting to exchange a witness for a digital coin held by the buyers. Fairness in this setting requires that all parties reach a consensus on the exchange; otherwise, no transaction takes place-ensuring that either all buyers simultaneously receive the witness or none do. In addition, fairness guarantees that no individual seller can unilaterally reveal the witness and that each seller either receives an equal share of the payment or no compensation at all.

3.1 Preliminaries

NOTATIONS. We denote uniform sampling of x from the set X by $x \leftarrow\!\$\ X$, and let λ represent the security parameter. For a probabilistic polynomial-time (PPT) algorithm A that, given input x and randomness r, outputs y, we write $y \leftarrow \mathsf{A}(x;r)$. If A is a deterministic polynomial-time (DPT) algorithm, we use the notation $y := \mathsf{A}(x)$. A function $\mathsf{negl}(\lambda) : \mathbb{N} \to \mathbb{R}$ is called negligible in λ if, for every $k \in \mathbb{N}$, there exists $\lambda_0 \in \mathbb{N}$ such that for all $\lambda \geq \lambda_0$, it holds that $|\mathsf{negl}(\lambda)| \leq \lambda^{-k}$. We use the function **assert** to enforce conditions. Specifically, the function call **assert** $\mathsf{condition}$ terminates the execution in which it is invoked if $\mathsf{condition}$ evaluates to false. A vector of elements $x_1, \ldots, x_n$ is denoted as $\mathbf{X}$. To denote the execution of a protocol among parties $\mathcal{S}_1, \ldots, \mathcal{S}_n$, we write $\langle \mathcal{S}_1(k_1, m), \ldots, \mathcal{S}_n(k_n, m) \rangle$. If the inputs of the parties are clear from the context, we omit them. When an adversary controls multiple parties, we use an abuse of notation to collapse all adversarial parties into a single entity, denoted as $\langle \mathcal{S}_1, \mathcal{A} \rangle$.

BLOCKCHAINS. This paper contains both positive and negative results that rely on blockchains with different capabilities. Our construction relies on transactions and a blockchain; we recall the formal definitions in this section. In Section 7, we show that it is generally impossible to achieve fairness for sellers in the presence of a minimal blockchain. To address this limitation, we introduce the notion of minimal blockchains with timelocks. Looking ahead, incorporating timelocks will allow us to build a MPFE protocol that ensures seller fairness. To model timelocks in a way that is compatible with blockchain systems such as Bitcoin, we follow the abstraction approach of [Aum+21].

Definition 1 (Transaction)**.** *A transaction* tx *is a tuple* $(\mathsf{pk}_\mathsf{S}, \mathsf{tx}_\mathsf{prev}, \mathbf{PK}, \mathbf{X})$ *consisting of a sender public key* pk_S*, an input transaction* $\mathsf{tx}_\mathsf{prev}$*, a vector of receiver public keys* $\mathbf{PK}$*, and a vector of receiver amounts* $\mathbf{X}$*.*

The first definition we present considers the blockchain as an immutable bulletin board [Suw+22], where transactions are appended sequentially and cannot be modified once recorded.

Definition 2 (Minimal Blockchain)**.** *A minimal blockchain* $\mathcal{C}$ *is a public bulletin board consisting of an ordered list of transactions. It provides the interfaces* post *and* bal*, which work like follows:*

$\mathcal{C}' \leftarrow \mathsf{post}(\mathcal{C}, tx, \sigma)$*: The post interface has as input a transaction* tx*, and a signature* σ*. The post interface outputs an updated chain* $\mathcal{C}'$*.*

$x \leftarrow \mathsf{bal}(\mathsf{pk})$*: The balance interface takes as input a public key* pk *and outputs the amount that is associated with the public key* pk *at the current state of the blockchain.*

A transaction-signature pair (tx, σ) that is posted on the blockchain $\mathcal{C}$ is valid if

- σ is a signature that verifies on the message tx and the public key pk, and
- the input transaction $\mathsf{tx}_{\mathsf{prev}}$ was never spent by any transaction signed by pk that is posted on the chain, and
- the outputs of the input transaction $\mathsf{tx}_{\mathsf{prev}}$ that correspond to pk equal the cumulative outputs of the transaction tx, and
- all receiver amounts are non-negative, i.e. $x_i \geq 0 \; \forall x_i \in \mathbf{X}$.

Each valid transaction-signature pair that is posted on the blockchain updates the balance of the sender and the recipient. The balance of each recipient $\mathsf{pk}_{\mathsf{R},i} \in \mathbf{PK}$ is increased by the amount x_i and the balance of the sender pk is decreased by $\sum_{x_i \in \mathbf{X}} x_i$. From this point forward, we assume that each honestly generated public key has a non-empty balance at the beginning of our protocol, i.e., there exists an initial transaction whose receivers are all honestly generated public keys. In addition, if a coin is owned by a public key whose corresponding signing key is shared by n parties, we add to the balance of each shareholder $1/n$ coins.

The next definition extends this minimal view of the blockchain by incorporating *timelocks*, which enforce time constraints on when transactions can be executed.

Definition 3 (Minimal Blockchain with Timelocks)**.** *A minimal blockchain has timelocks if it provides the* lock *interface.*

$\mathcal{C}' \leftarrow \mathsf{lock}(\mathcal{C}, \sigma, tx, R, \mathtt{t})$*: The lock interface has as input a signature* σ*, a transaction* tx*, a commitment* R*, and a time* $\mathtt{t}$*. It updates an updated chain* $\mathcal{C}'$*.*

We define a transaction as *locked* if σ is a valid signature on $(\mathsf{tx}, R, \mathtt{t})$ with respect to the signing key pk that controls the outputs of tx, and fewer than $\mathtt{t}$ messages have been posted to the blockchain since the lock interface was invoked. While alternative measures of time can be used, such as parties signing transactions to mark the end of rounds, for our proof it is sufficient that the adversary cannot delay messages beyond the duration of the lock. Thus, counting the number of messages posted is sufficient for our purposes. When a transaction is locked, it can only be used as a valid input for a payment if the signature authorizing that payment matches the commitment R. In our construction we use Schnorr signatures, i.e., a transaction can be spent if the signature is of the form (R, z). However, the commitment can be chosen arbitrarily to regulate payments before the lock expires. To extend the lock notation to a multiparty setting, we denote the locking operation as $\mathcal{C}.\mathsf{lock}(\mathsf{sk}_i, \mathsf{tx}, R)$, indicating that the signers jointly generate the locking signature using their respective signing keys. In addition, we omit explicit mention of the locking duration, since it is determined as a protocol parameter based on the adversary's delay capabilities.

3.2 Fair Exchange

A multiparty fair exchange (MPFE) protocol is executed between a set of n_{S} sellers and n_{B} buyers. For simplicity, we assume that the parties agree on the exchange terms before running the protocol. By agreement, sellers jointly sell a witness w to buyers for a publicly known statement s. Fairness

requires that either both the witness and the payment are exchanged, or neither occurs. To enforce fairness, w is secretly shared among sellers using a threshold scheme SShare, which ensures that reconstruction requires the cooperation of a sufficient number of sellers. Let w_j denote the share of w associated with s. Similarly, buyers collectively control the payment. This is formalized by distributing shares of the signing key $\mathsf{sk}_{\mathsf{B}_i}$ to each buyer, where $1 \leq i \leq n_{\mathsf{B}}$. Upon agreement to sell, the buyers collaboratively generate a signature that verifies under the public key pk_{B}, which is associated with the payment transaction.

Definition 4 (Fair Distributed Buying)**.** *Let $\mathcal{R}$ be a hard relation and $\mathcal{C}$ be a blockchain. A* fair distributed buying $\mathsf{MPFE}_{\mathcal{R},\mathcal{C}}$ *of a coin owned by a public key pk_{B}, for a statement-witness pair (s, w) of the hard relation $\mathcal{R}$, where the witness w is shared into n_{S} parts via $\mathsf{SShare}(w)$ into $(w_1, \ldots, w_{n_{\mathsf{S}}})$, executed between a set of buyers B_i $(1 \leq i \leq n_{\mathsf{B}})$ and a set of sellers S_j $(1 \leq j \leq n_{\mathsf{S}})$ consists of the algorithm Setup and the protocol MPFE which are defined as follows:*

$(\mathsf{pk}_{\mathsf{B}}, \mathsf{sk}_{\mathsf{B}_1}, \ldots, \mathsf{sk}_{\mathsf{B}_{n_{\mathsf{B}}}}, \mathsf{sk}_{\mathsf{S}_1}, \ldots, \mathsf{sk}_{\mathsf{S}_{n_{\mathsf{S}}}}) \leftarrow \mathsf{Setup}(t_{\mathsf{B}}, n_{\mathsf{B}}, t_{\mathsf{S}}, n_{\mathsf{S}})$*: The* setup *algorithm takes as input the thresholds $(t_{\mathsf{B}}, t_{\mathsf{S}})$, and the number of participants $(n_{\mathsf{B}}, n_{\mathsf{S}})$. It outputs a public key pk_{B}, a secret key $\mathsf{sk}_{\mathsf{B}_i}$ for every buyer, and a secret key $\mathsf{sk}_{\mathsf{S}_j}$ for each seller.*

$(w', \perp) \leftarrow \langle \{\mathsf{B}_i^{\mathcal{C}}(\mathsf{sk}_{\mathsf{B}_i}, s, \mathsf{tx})_i\}_{i \in \mathcal{B}}, \{\mathsf{S}_j^{\mathcal{C}}(\mathsf{sk}_{\mathsf{S}_j}, \mathsf{tx}, s, w_j)_j\}_{j \in \mathcal{S}} \rangle_{\mathsf{MPFE}}$*: The* MPFE *protocol is executed between n_{B} buyers and n_{S} sellers. Each buyer B_i has as input its signing key $\mathsf{sk}_{\mathsf{B}_i}$, a statement s, and a transaction tx. Each seller has as input a secret key $\mathsf{sk}_{\mathsf{S}_j}$, a transaction tx, a statement s, and a partial witness w_j. In addition, each party has access to a blockchain $\mathcal{C}$. The protocol outputs a witness w' for the buyers and nothing for the sellers.*

A fair distributed buying protocol ($\mathsf{MPFE}_{\mathcal{R},\mathcal{C}}$) is correct if, for any security parameter λ, statement-witness pair (s, w), all keys $\mathsf{pk}_{\mathsf{B}}, \mathsf{sk}_{\mathsf{B}_i}, \mathsf{sk}_{\mathsf{S}_j}$ and partial witnesses output by $w_1, \ldots, w_{n_{\mathsf{S}}} \leftarrow \mathsf{SShare}(w)$ generated by $\mathsf{Setup}(t_{\mathsf{B}}, n_{\mathsf{B}}, t_{\mathsf{S}}, n_{\mathsf{S}})$, for every transaction tx, it holds true that: For each each witness w' output by MPFE, the condition $\mathcal{R}(s, w') = 1$ is satisfied. In addition, the blockchain $\mathcal{C}$ contains a signature σ_{coin} that verifies w.r.t. the buyer's combined public key pk_{B} on a valid transaction from pk_{B} to all sellers.

3.3 Security Properties

We define the security properties of an MPFE protocol through *buyer fairness*, *seller fairness*, and *witness privacy*, beginning with the adversarial model.

Adversarial model. Our security analysis considers a static adversary, meaning that corruptions occur before the adversary interacts with the protocol [Chu+23]. This assumption is widely used in fairness research, particularly in two-party settings where corruption decisions are fixed at the outset. While some of our building blocks achieve adaptive security, extending our analysis to an adaptive corruption model introduces significant technical challenges and is beyond the scope of this work.

Buyer Fairness. Buyer fairness protects the honest buyers. Buyer fairness guarantees that if an adversary controls all sellers and corrupts up to $t_{\mathsf{B}} - 1$ buyers, the adversary only gets paid if all honest buyers can reveal a valid witness.

Definition 5 (Buyer Fairness)**.** *Let $\mathcal{R}$ be a hard relation and $\mathcal{C}$ be a blockchain. A multi-party fair exchange protocol* ($\mathsf{MPFE}_{\mathcal{R},\mathcal{C}}$) *achieves* t_{B}-buyer fairness*, if for every* PPT *adversary $\mathcal{A}$ there exists*

a negligible function negl *such that the probability*

$$\Pr\left[\mathsf{BF}^{\mathcal{A}}_{\mathsf{TAS},t_\mathsf{B},n_\mathsf{B},n_\mathsf{S}}(\lambda)=1\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{BF}^{\mathcal{A}}_{\mathsf{TAS},t_\mathsf{B},n_\mathsf{B},n_\mathsf{S}}$ *is described in Figure 1, the probability is taken over the random choices of all probabilistic algorithms, and* $t_\mathsf{B} \leq n_\mathsf{B} \leq \lambda$.

$\underline{\mathsf{BF}^{\mathcal{A}}_{\mathsf{TAS},t_\mathsf{B},n_\mathsf{B},n_\mathsf{S}}(\lambda)}$:

1. The challenger runs the setup to generate the secret keys of the buyers and sellers. In addition, the challenger samples a statement-witness pair (s, w) of the relation $\mathcal{R}$.
2. The challenger forwards all public keys and the secret keys of all sellers and the statement-witness pair (s, w) to $\mathcal{A}$, and $\mathcal{A}$ selects the set of corrupted buyers corr_B with a size of at most $t_\mathsf{B} - 1$.
3. The challenger forwards the secret key of each corrupted buyer to $\mathcal{A}$, and $\mathcal{A}$ can interact with all honest buyers.
4. Eventually, $\mathcal{A}$ finishes the game.
5. The adversary wins the buyer fairness game if
   - there is a valid payment on the chain $\mathcal{C}$, which spends a coin from the buyers; and
   - at least one honest buyer B_i cannot output a valid witness for the statement s.

Figure 1: Security game for buyer fairness.

Seller Fairness. The seller fairness security property considers the case where all buyers are malicious, and the attacker additionally controls up to $t_\mathsf{S} - 1$ sellers. Seller fairness guarantees in this setting that if the adversary learns the witness, then all honest sellers are compensated.

To capture this intuitive security goal, a natural first step is to require the adversary to output a valid witness and a signature that verifies w.r.t. pk_B on a transaction for which at least one honest seller is not a recipient. Intuitively, one could argue that this signature causes a payment in which at least one honest seller is not the recipient. Yet, we cannot bind the winning condition to the knowledge of valid signatures due to a concurrency problem, which we call *frontrunning*.

Frontrunning. As a frontrunning adversary, we describe an adversary that follows a protocol honestly, i.e., sends the outputs to all other parties but does concurrent actions, which makes the protocol outputs irrelevant. In the case of seller fairness, frontrunning happens if the adversary $\mathcal{A}$ acts as both a seller and a buyer. Then, if $\mathcal{A}$ learns the signature on a transaction first, it can pause the protocol and spend the inputs of the transaction *before* continuing the protocol. Therefore, frontrunning leads to an adversary that follows the protocol honestly and yet manipulates the outcome of the protocol into a not-desired one: While each seller learns a signature on a transaction tx, the signature does not correspond to a valid payment since the inputs of tx have already been

spent. To mitigate this unfortunate state of affairs, we utilize a blockchain $\mathcal{C}$ and compare the balance of each honest seller before and after the execution of the protocol. This enforces that the adversary not only outputs a signature (which might not lead to a payment) but actually makes a payment, which is desired for seller fairness.

Comparing Balances. The remaining question is, which condition does the adversary need to satisfy to break seller fairness? In exchange for the interaction with the adversary, the honest seller expects to receive a payment worth its share of the witness value, which is $1/n_{\mathsf{S}}$. So we challenge the adversary to output a valid witness that is initially shared amongst the sellers while ensuring that when the game terminates, at least one honest seller has a final balance $\mathsf{bal}_{\mathsf{post}}$ of less than $\mathsf{bal}_{\mathsf{post}} < \mathsf{bal}_{\mathsf{prev}} + \frac{1}{n_{\mathsf{S}}}$. Additionally, we extend the winning condition to account for scenarios where the adversary does not learn the witness, yet at least one honest seller ends up with fewer funds than before the protocol execution. This ensures that honest parties are protected from financial loss and aligns with the guarantees provided by buyer fairness.

Definition 6 (Seller Fairness)**.** *Let $\mathcal{R}$ be a hard relation and $\mathcal{C}$ be a blockchain. A multi-party fair exchange protocol* ($\mathsf{MPFE}_{\mathcal{R},\mathcal{C}}$) *achieves* t_{S}-seller fairness*, if for every* PPT *adversary $\mathcal{A}$ there exists a negligible function* negl *such that the probability*

$$\Pr\left[\mathsf{SF}^{\mathcal{A}}_{\mathsf{TAS},n_{\mathsf{B}},t_{\mathsf{S}},n_{\mathsf{S}},\mathcal{C}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{SF}^{\mathcal{A}}_{\mathsf{TAS},n_{\mathsf{B}},t_{\mathsf{S}},n_{\mathsf{S}},\mathcal{C}}$ is described in Figure 2, the probability is taken over the random choices of all probabilistic algorithms, $t_{\mathsf{S}} \leq n_{\mathsf{S}} \leq \lambda$, and $n_{\mathsf{B}} \leq \lambda$.

Witness Privacy.

Witness privacy ensures that w is disclosed only to buyers and remains hidden from sellers who do not know it before protocol execution. Since w is secretly shared among n_{S} sellers, requiring at least t_{S} shares for reconstruction, the definition assumes that less than t_{S} malicious sellers cannot reconstruct w after execution. To prevent trivial leakage, we assume that buyers act honestly, since they eventually learn w and might otherwise leak it to malicious sellers.

Definition 7 (Witness Privacy)**.** *Let $\mathcal{R}$ be a hard relation and $\mathcal{C}$ be a blockchain. A multi-party fair exchange protocol* ($\mathsf{MPFE}_{\mathcal{R},\mathcal{C}}$) *achieves* t_{S}-witness privacy*, if for every* PPT *adversary $\mathcal{A}$ there exists a negligible function* negl *such that the probability*

$$\Pr\left[\mathsf{WitPriv}^{\mathcal{A}}_{\mathsf{TAS},n_{\mathsf{B}},t_{\mathsf{S}},n_{\mathsf{S}},\mathcal{C}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{WitPriv}^{\mathcal{A}}_{\mathsf{TAS},n_{\mathsf{B}},t_{\mathsf{S}},n_{\mathsf{S}},\mathcal{C}}(\lambda)$ is described in Figure 3, the probability is taken over the random choices of all probabilistic algorithms, $t_{\mathsf{S}} \leq n_{\mathsf{S}} \leq \lambda$, and $n_{\mathsf{B}} \leq \lambda$.

Flexibility of our setting. Our security definitions account for both scenarios where each seller is paid individually and where a single transaction transfers funds from the buyers to a shared key controlled by the sellers. Additionally, standard techniques allow for modifying our protocol so that each buyer learns only a share of the witness rather than the full witness.

4 Threshold Adaptor Signatures

We introduce *threshold adaptor signatures* (TAS), a generalization of classical adaptor signatures that allows a subset of signers to jointly compute a pre-signature. Unlike standard adaptor signa-

$\underline{\mathsf{SF}^{\mathcal{A}}_{\mathsf{TAS},n_\mathsf{B},t_\mathsf{S},n_\mathsf{S},\mathcal{C}}(\lambda)}$:

1. The challenger runs the setup to generate the secret keys of the buyers and sellers and stores the balance of each public key into the list $\mathsf{bal}_{\mathsf{prev}}$.
2. The challenger forwards all public keys and the secret keys of all buyers to $\mathcal{A}$, and $\mathcal{A}$ selects the set of corrupted sellers corr_S with a size of at most $t_\mathsf{S} - 1$.
3. The challenger forwards the secret key of each corrupted seller to $\mathcal{A}$.
4. The challenger samples a fresh statement-witness pair (s, w) from the relation $\mathcal{R}$, and shares the witness w into n_S parts running the SShare algorithm. The challenger outputs the statement s and all witness shares that correspond to malicious sellers to $\mathcal{A}$ and stores the witness shares of the honest sellers.
5. In addition, $\mathcal{A}$ can interact with all honest sellers.
6. Eventually, $\mathcal{A}$ outputs a witness w', and the challenger stores the balance of each public key in the list $\mathsf{bal}_{\mathsf{post}}$.
7. The adversary wins if
   - the witnesses w' is a valid witness for the challenge statement s; and
   - there exists at least an honest seller S_j, whose balance after the interaction with the adversary is smaller than the expected value, i.e.,

     $$\mathsf{bal}_{\mathsf{post}}[\mathsf{S}_j] - \mathsf{bal}_{\mathsf{prev}}[\mathsf{S}_j] < \frac{1}{n_\mathsf{S}},$$

     or, the adversary also wins the seller fairness game without learning the witness if there exists an honest seller whose balance after the interaction with the adversary is smaller than before the interaction, i.e.,

     $$\mathsf{bal}_{\mathsf{post}}[\mathsf{S}_j] < \mathsf{bal}_{\mathsf{prev}}[\mathsf{S}_j].$$

Figure 2: Security game for seller fairness.

tures, where the pre-signature is derived via a single algorithm, our threshold construction requires an interactive protocol between at least t-out-of-n signers to produce a pre-signature $\widetilde{\sigma}$.

To ensure compatibility with the underlying blockchain, the resulting pre-signature must be indistinguishable from a single-user pre-signature. Furthermore, our scheme preserves the fundamental properties of classical adaptor signatures:

- **Adaptation:** Given a pre-signature $\widetilde{\sigma}$ and a witness w, one can compute a full signature σ.

$\underline{\mathsf{WitPriv}^{\mathcal{A}}_{\mathsf{TAS},n_\mathsf{B},t_\mathsf{S},n_\mathsf{S},\mathcal{C}}(\lambda)}$:

1. The challenger runs the setup to generate the secret keys of the buyers and sellers.
2. The challenger forwards the public key pk_B to $\mathcal{A}$, and $\mathcal{A}$ selects the set of corrupted sellers corr_S that has size of at most $t_\mathsf{S} - 1$.
3. The challenger forwards the secret key of each corrupted seller to $\mathcal{A}$.
4. The challenger samples a fresh statement-witness pair (s, w) from the relation $\mathcal{R}$, and shares the witness w into n_S parts running the SShare algorithm. The challenger forwards the statement s and all witness shares that correspond to malicious sellers to $\mathcal{A}$ and stores the witness shares of the honest sellers.
5. $\mathcal{A}$ can interact with all honest sellers and all buyers.
6. Eventually, $\mathcal{A}$ outputs a witness w^*.
7. The adversary wins the witness privacy game if the witness w^* is a valid witness for the statement s.

Figure 3: Security game for $\mathsf{WitPriv}$.

- **Extraction:** Given a pre-signature $\widetilde{\sigma}$, the adapted pre-signature σ, and a statement s, one can extract the corresponding witness w.

4.1 Defining Threshold Adaptor Signatures

Threshold adaptor signatures are defined as follows:

Definition 8 (Threshold Adaptor Signature)**.** *A* threshold adaptor signature scheme *w.r.t. a hard relation $\mathcal{R}$ for some language $L_\mathcal{R}$ and a threshold signature scheme* $\mathsf{TS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *consists of a tuple of four algorithms and protocols* $\mathsf{TAS}_{\mathcal{R},\mathsf{TS}} = (\mathsf{pSign}, \mathsf{Adapt}, \mathsf{pVrfy}, \mathsf{Extract})$ *defined as:*

$\underline{\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, \mathsf{pk}, m, s) \rangle}$. *The* pre-signing protocol *is an interactive algorithm of which an instance is run by each signer $\mathcal{S}_1, \ldots, \mathcal{S}_t$ concurrently. Concretely, signer $\mathcal{S}_i$ runs pSign_i, which takes as input a secret key share sk_i, the joined public key pk, a message m, and a NP-statement s. At the end of the protocol, $\mathcal{S}_i$ obtains a pre-signature $\widetilde{\sigma}$ as output.*

$\underline{b \leftarrow \mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma})}$. *The* pre-verification algorithm *is a* DPT *algorithm that on input a public key* pk*, message $m \in \{0,1\}^{\ell_m}$, statement $s \in L_\mathcal{R}$, and pre-signature $\widetilde{\sigma}$, outputs a bit b.*

$\underline{\sigma \leftarrow \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w)}$. *The* adapting algorithm *is a* DPT *algorithm that on input a pre-signature $\widetilde{\sigma}$ and witness w for the statement $s \in L_\mathcal{R}$ outputs an adapted signature σ.*

$\underline{w \leftarrow \mathsf{Extract}(\mathsf{pk}, \widetilde{\sigma}, \sigma, s)}$. *The* extracting algorithm *is a* DPT *algorithm that on input a pre-signature*

$\widetilde{\sigma}$, signature σ, and statement $s \in L_{\mathcal{R}}$, outputs a witness w such that $(s, w) \in \mathcal{R}$, or $\perp$.

A threshold adaptor signature is correct under the following intuitive conditions: First, for any honestly generated set of keys and honestly computed instance of some hard relation, any pre-signature that has been honestly generated and which is denoted by $\widetilde{\sigma}$, the adaption of $\widetilde{\sigma}$ for a corresponding witness w, should result in a valid signature σ, using the specified adaptation algorithms. Furthermore, given both a valid pre-signature and a valid signature together with the statement s, it should be computationally feasible to deduce or extract the corresponding witness w efficiently. We refer to this property as *pre-signature correctness*.

Definition 9 (Pre-signature correctness)**.** *A threshold adaptor signature scheme* TAS *satisfies* pre-signature correctness*, if for all $\lambda, t < n \in \mathbb{N}$ and $m \in \{0,1\}^{\ell_m}$*

$$\Pr\left[\begin{array}{l|l} \mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma}) = 1 \wedge & par \leftarrow \mathsf{Setup}(n, t) \\ \mathsf{Vrfy}(\mathsf{pk}, m, \sigma) = 1 \wedge & (\mathsf{pk}, \mathsf{sk}_i) \leftarrow \langle \mathsf{KeyGen}(i) \rangle \\ (s, w') \in \mathcal{R} & (s, w) \leftarrow \mathsf{genR}(1^\lambda), \\ & \widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, \mathsf{pk}, m, s) \rangle \\ & \sigma := \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w), \\ & w' := \mathsf{Extract}(\mathsf{pk}, \widetilde{\sigma}, \sigma, s) \end{array}\right] = 1.$$

4.2 Security of Threshold Adaptor Signatures

We adapt the security definitions of Gerhart et al.[Ger+24a] to the threshold setting. A threshold adaptor signature scheme must satisfy extractability, unique extractability, unlinkability, pre-signature adaptability, and pre-verification soundness. These properties are formally defined below.

Extractability. Extractability ensures that a malicious party cannot use a pre-signature $\widetilde{\sigma}$ w.r.t. a statement s to forge a valid signature σ without revealing a corresponding witness w, thereby protecting the signer.

Definition 10 (Extractability)**.** *A threshold adaptor signature scheme* TAS *is* extractable *if for every* PPT *adversary $\mathcal{A}$, there is a negligible function* negl *such that for every $\lambda, t < n \in \mathbb{N}$*

$$\Pr\left[\mathsf{Ext}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{Ext}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$ is defined in Figure 4 and the randomness is taken over all random coins used by the probabilistic algorithms.

Unique Extractability. Unique extractability guarantees that a verified pre-signature commits to a single valid signature and a single witness. No efficient adversary can generate a pre-signature $\widetilde{\sigma}$ for a message m and a statement s such that two distinct signatures on m can be derived from $\widetilde{\sigma}$, each revealing a valid witness. More formally:

Definition 11 (Unique Extractability)**.** *A threshold adaptor signature scheme* TAS *is unique extractable, if for every* PPT *adversary $\mathcal{A}$ there exists a negligible function* negl *such that for every $\lambda, t < n \in \mathbb{N}$*

$$\Pr\left[\mathsf{UniqueExtractability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda)\ ,$$

where experiment $\mathsf{Ext}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$ is described in Figure 5, and the probability is taken over the random choices of all probabilistic algorithms.

$\mathsf{Ext}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$

$par \leftarrow \mathsf{Setup}(n,t)$
$(\mathsf{pk}, \{\mathsf{sk}_j\}_{j\in\{1,\dots,n\}}) \leftarrow \mathsf{KeyGen}(par)$
$\mathsf{corr} \leftarrow \mathcal{A}(\mathsf{pk})$; $\mathsf{hon} \leftarrow \{1,\dots,n\} \setminus \mathsf{corr}$
$\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}}, \mathcal{Q}_{\mathsf{stmt}} \leftarrow \emptyset$
$\mathbb{O} := (\mathsf{pSign}(\cdot,\cdot,\cdot), \mathsf{Sign}(\cdot,\cdot), \mathsf{NewS}())$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathbb{O}}(\mathsf{pk}, \{\mathsf{sk}_j\}_{(j\in\mathsf{corr})}))$
assert $\mathsf{Vrfy}(\mathsf{pk}, m, \sigma^*)$
assert $m^* \notin \mathcal{Q}_{\mathsf{Sign}}$
// The adversary cannot succeed without disclosing shares
assert $\forall(\widetilde{\sigma}, s) \in \mathcal{Q}_{\mathsf{pSign}}[m^*] : \mathsf{pVrfy}(\mathsf{pk}, \widetilde{\sigma}, s)$
return $(\forall(\widetilde{\sigma}, s) \in \mathcal{Q}_{\mathsf{pSign}}[m^*]$ s.t. $s \notin \mathcal{Q}_{\mathsf{stmt}}$:
    $(s, \mathsf{Extract}(s, \widetilde{\sigma}, \sigma^*)) \notin \mathcal{R})$

$\mathsf{pSign}(i, m, s)$

assert $i \in \mathsf{hon}$
$\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, m, s), \mathcal{A} \rangle$
$\mathcal{Q}_{\mathsf{pSign}}[m] \xleftarrow{\cup} \{(\widetilde{\sigma}, s)\}$
return $\widetilde{\sigma}$

$\mathsf{Sign}(i, m)$

assert $i \in \mathsf{hon}$
$\sigma \leftarrow \langle \mathsf{Sign}(\mathsf{sk}_i, m), \mathcal{A} \rangle$
$\mathcal{Q}_{\mathsf{Sign}} \xleftarrow{\cup} \{m\}$
return σ

$\mathsf{NewS}(\lambda)$

$(s, w) \leftarrow \mathcal{R}.\mathsf{genR}(1^\lambda)$
$\mathcal{Q}_{\mathsf{stmt}} \xleftarrow{\cup} s$
return s

Figure 4: The security game $\mathsf{Ext}^{\mathcal{A}}_{\mathsf{TAS},t,n,\mathsf{corr}}(\lambda)$.

Unlinkability. Unlinkability ensures that an adversary cannot distinguish between standard signatures and those derived from adapted pre-signatures, even if the adaptations use adversary-provided witnesses. Since adversary-controlled signers can trivially determine whether a signature was obtained by signing or by pre-signing and adapting, we depart from the notion of unlinkability for non-threshold adaptor signatures. Instead, we adopt the canonical signing notion from [DOY22] to define unlinkability in the threshold setting. Intuitively, a threshold adaptor signature scheme achieves unlinkability if adapted pre-signatures are indistinguishable from ordinary signatures.

Definition 12 (Unlinkability)**.** *A threshold adaptor signature scheme* TAS *is unlinkable if for every two-stage* PPT *adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *there exists a negligible function* negl *such that for every* $\lambda, t < n \in \mathbb{N}$

$$\Pr\left[\mathsf{Unlinkability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda) \ ,$$

where experiment $\mathsf{Unlinkability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$ *is described in Figure 6, and the probability is taken over the random choices of all probabilistic algorithms.*

Pre-Signature Adaptability. Pre-signature adaptability ensures that a pre-signature related to a statement s can be transformed into a valid signature with a witness w, even in cases where the signer acts maliciously.

Definition 13 (Pre-signature adaptability)**.** *A threshold adaptor signature scheme* TAS *satisfies* pre-signature adaptability*, if for all* $\lambda \in \mathbb{N}$*, messages and pre-signatures* $\widetilde{\sigma}, m \in \{0,1\}^*$*, statement/witness pairs* $(s, w) \in \mathcal{R}$*, public keys* pk *we have,*

$$\textit{if } \mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma}) = 1, \ \textit{then } \mathsf{Vrfy}(\mathsf{pk}, m, \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w)) = 1.$$

$\mathsf{UniqueExtractability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$

$par \leftarrow \mathsf{Setup}(n,t)$
$(\mathsf{pk}, \{\mathsf{sk}_j\}_{j\in\{1,\ldots,n\}}) \leftarrow \mathsf{KeyGen}(par)$
$\mathsf{corr} \leftarrow \mathcal{A}(\mathsf{pk});\ \mathsf{hon} \leftarrow \{1,\ldots,n\} \setminus \mathsf{corr}$
$\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}}, \mathcal{Q}_{\mathsf{stmt}} \leftarrow \emptyset$
$\mathbb{O} := (\mathsf{pSign}(\cdot,\cdot,\cdot), \mathsf{Sign}(\cdot,\cdot))$
$(m, s, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathbb{O}}(\mathsf{pk}, \{\mathsf{sk}_j\}_{(j\in\mathsf{corr})})$
assert $(\sigma \neq \sigma')$
assert $\mathsf{Vrfy}(\mathsf{pk}, m, \sigma) \wedge \mathsf{Vrfy}(\mathsf{pk}, m, \sigma')$
assert $\mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma})$
$y \leftarrow \mathsf{Extract}(s, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(s, \widetilde{\sigma}, \sigma')$
return $(s,w) \in \mathcal{R} \wedge (s,w') \in \mathcal{R}$

$\mathsf{pSign}(i, m, s)$

assert $i \in \mathsf{hon}$
$\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, m, s), \mathcal{A} \rangle$
return $\widetilde{\sigma}$

$\mathsf{Sign}(i, m)$

assert $i \in \mathsf{hon}$
$\sigma \leftarrow \langle \mathsf{Sign}(\mathsf{sk}_i, m), \mathcal{A} \rangle$
return σ

Figure 5: The security game $\mathsf{UniqueExtractability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$.

Pre-Verify Soundness. Pre-verify soundness ensures that the pre-verification algorithm satisfies *computational* soundness w.r.t. the relation $\mathcal{R}$. In particular, pVrfy should reject pre-signatures computed using statements $s \notin \mathcal{R}$. Intuitively, pre-verify soundness ensures that every valid pre-signature can be adapted to a full signature, and one can extract a witness from it. This strengthens the property of pre-signature adaptibility (Definition 13), which is restricted to honestly generated pre-signatures on statements in the relation.

Definition 14 (Computational Pre-Verify Soundness)**.** *An adaptor signature scheme* TAS *satisfies* computational pre-verify soundness *if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* negl *such that for every* $\lambda \in \mathbb{N}$ *and polynomially-bounded* $s \notin \mathcal{L}_{\mathcal{R}}$*,*

$$\mathsf{Pr}\left[(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(\lambda), (m, \widetilde{\sigma}) \leftarrow \mathcal{A}(\mathsf{sk}) : \mathsf{pVrfy}(\mathsf{pk}, m, \widetilde{\sigma}, s) = 1\right] \leq \nu(\lambda).$$

4.3 Schnorr Threshold Adaptor Construction

We present a Schnorr threshold adaptor signature scheme and refer the reader to Section 2.1 for an outline of the construction.

Construction 1 (Schnorr Threshold Adaptor Signature)**.** *Let* $\lambda \in \mathbb{N}$ *be the security parameter,* $\mathcal{R}$ *be a relation for the language* $\mathcal{L}_{\mathcal{R}} := \{(s,w) | s = g^w\}$*. Let* $\mathsf{Sparkle}$ *be a threshold signature as defined in Construction 3. We construct the threshold adaptor signature* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *by defining the algorithms* $\mathsf{pSign}^0, \mathsf{pSign}^1, \mathsf{pSign}^2, \mathsf{Combine}, \mathsf{Adapt}, \mathsf{Extract}, \mathsf{pVrfy}$ *in Figure 7 which lead to the following pre-signing protocol.*

$\underline{\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, \mathsf{pk}, m, s) \rangle}$ *The pre-signing protocol constists of the three pre-signing rounds* pSign^0, $\mathsf{pSign}^1, \mathsf{pSign}^2$*, and a combination algorithm* $\mathsf{Combine}$*. Each signer executes the pre-signing rounds, and the local outputs* $\rho_i, \rho_i', \rho_i''$ *of each pre-signing round are forwarded to all other signers. After the three rounds are executed, each signer combines the outputs* $\{\rho_i\}_{i\in\mathcal{S}}$ *using the* $\mathsf{Combine}$ *algorithm.*

$\mathsf{Unlinkability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$

1 : $par \leftarrow \mathsf{Setup}(n,t)$; parse $\mathcal{A}$ as $(\mathcal{A}_1, \mathcal{A}_2)$
2 : $(\mathsf{pk}, \{\mathsf{sk}_j\}_{j\in\{1,\dots,n\}}) \leftarrow \mathsf{KeyGen}(par)$
3 : $\mathsf{corr} \leftarrow \mathcal{A}_1(\mathsf{pk})$; $\mathsf{hon} \leftarrow \{1,\dots,n\} \setminus \mathsf{corr}$
4 : $\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}}, \mathcal{Q}_{\mathsf{stmt}} \leftarrow \emptyset$; $b \leftarrow\$ \{0,1\}$
5 : $\mathbb{O} := (\mathsf{pSign}(\cdot,\cdot,\cdot), \mathsf{Sign}(\cdot,\cdot), \mathsf{Chall}(b,\cdot,\cdot,\cdot))$;
6 : $\mathcal{A}_1(\{\mathsf{sk}_j\}_{(j\in\mathsf{corr})}))$; $b' \leftarrow \mathcal{A}_2^{\mathbb{O}}(\mathsf{pk}, \mathsf{hon})$
7 : **return** $b' == b$

$\mathsf{Chall}(b, i, m, (s, w))$

1 : **assert** $i \in \mathsf{hon}$
2 : **assert** $(s, w) \in \mathcal{R}$
3 : $\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, m, s), \mathcal{A}_1 \rangle$
4 : $\sigma_0 \leftarrow \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w)$
5 : $\sigma_1 \leftarrow \langle \mathsf{Sign}(\mathsf{sk}_i, m), \mathcal{A}_1 \rangle$
6 : **return** σ_b

$\mathsf{pSign}(i, m, s)$

1 : **assert** $i \in \mathsf{hon}$
2 : $\widetilde{\sigma} \leftarrow \langle \mathsf{pSign}(\mathsf{sk}_i, m, s), \mathcal{A}_1 \rangle$
3 : **return** $\widetilde{\sigma}$

$\mathsf{Sign}(i, m)$

1 : **assert** $i \in \mathsf{hon}$
2 : $\sigma \leftarrow \langle \mathsf{Sign}(\mathsf{sk}_i, m), \mathcal{A}_1 \rangle$
3 : **return** σ

Figure 6: The security game $\mathsf{Unlinkability}^{\mathcal{A}}_{\mathsf{TAS},t,n}(\lambda)$.

$\mathsf{pSign}^0(i, m, s, \mathcal{S})$

1 : $r_i \leftarrow\$ \mathbb{Z}_p; R_i \leftarrow g^{r_i}$
2 : $\mathsf{cm}_i \leftarrow \mathsf{H}_{\mathsf{cm}}(m, \mathcal{S}, R_i, \boxed{s})$
3 : $\rho_i \leftarrow \mathsf{cm}_i$
4 : $\mathsf{st}_i \leftarrow (\rho_i, R_i, r_i, m, \boxed{s}, \mathcal{S})$
5 : **return** (ρ_i, st_i)

$\mathsf{pSign}^1(\mathsf{st}_i, \{\rho_i\}_{i\in\mathcal{S}}, \mathcal{S})$

1 : parse ρ_i as cm_i $\forall i \in \mathcal{S}$
2 : parse st_i as $(\rho_i, R_i, r_i, m, \boxed{s}, \mathcal{S})$
3 : **return** $\perp$ **if** $\mathsf{cm}_i \notin \{\mathsf{cm}_i\}_{i\in\mathcal{S}}$
4 : $\rho'_i \leftarrow R_i$
5 : $\mathsf{st}_i \leftarrow (\rho_i, R_i, r_i, m, \boxed{s}, \mathcal{S}, \{\rho_i\}_{i\in\mathcal{S}})$
6 : **return** (ρ'_i, st_i)

$\mathsf{pSign}^2(\mathsf{st}_i, \mathsf{sk}_i, \{\rho'_i\}_{i\in\mathcal{S}}, \mathcal{S})$

1 : // pSign^2 must be called once per st_i
2 : parse st_i as $(\rho_i, R_i, r_i, m, \boxed{s}, \mathcal{S}, \{\rho_i\}_{i\in\mathcal{S}})$
3 : parse ρ'_j as $R_j \forall j \in \mathcal{S}$
4 : **return** $\perp$ **if** $R_i \notin \{R_j\}_{j\in\mathcal{S}}$
5 : **for** $j \in \mathcal{S}$ **do**
6 : **return** $\perp$ **if** $\mathsf{cm}_j \neq \mathsf{H}_{\mathsf{cm}}(m, \mathcal{S}, R_j, \boxed{s})$
7 : $\widetilde{R} \leftarrow \prod_{i\in\mathcal{S}} R_i \boxed{\cdot s}$
8 : $h \leftarrow \mathsf{H}_{\mathsf{Sign}}(\mathsf{pk}, m, \widetilde{R})$
9 : $z_i \leftarrow r_i + h \cdot \lambda_i \cdot \mathsf{sk}_i$
10 : // λ_i is the Lagrange coefficient for i w.r.t. $\mathcal{S}$
11 : $\rho''_i \leftarrow z_i$
12 : **return** ρ''_i

Figure 7: Schnorr threshold adaptor signature construction based on Sparkle [CKM23]. We defer the algorithms Adapt, Extract, pVrfy, and Combine to Figure 13 in Appendix C. We mark the main changes compared to Sparkle in *gray*.

Theorem 1 (Security of Construction 1)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. If* Sparkle *is existentially unforgeable and* $\mathcal{R}$ *is a hard relation, then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves extractability, unique extractability, unlinkability, pre-signature adaptability, and computational pre-verify soundness in the random oracle model.*

We prove Theorem 1 by showing each security property in a separate lemma.

Lemma 1 (Extractability)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. If* Sparkle *is strongly unforgeable and* $\mathcal{R}$ *is a hard relation, then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves extractability in the random oracle model.*

Proof of Lemma 1. To prove Lemma 1, we follow the strategy of [Aum+21], which proves single-signer adaptor signatures secure. I.e., we build a reduction from the extractability of $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ to the hardness of the relation $\mathcal{R}$ and the strong unforgeability of Sparkle. The most challenging part of this reduction is to simulate the pre-signature oracle to the adversary, which we do as follows: When the adversary queries the pSign oracle for the signer i, the reduction calls its Sign^0 oracle and forwards the output to the adversary. When the adversary sends all of its commitments to the reduction, the reduction calls its Sign^1 oracle to receive its own commitment R_i. Afterward, the reduction extracts the commitments $\{R_j\}_{j\in\mathcal{S}\setminus i}$ of all signers by checking the inputs to the random oracle H_{cm}. Having these commitments available, the reduction queries its Sign^2 oracle and stores the partial signature $z_i \leftarrow r_i + h\lambda_i \mathsf{sk}_i$, which is output by its Sign^2 oracle. The reduction proceeds by programming the random oracle $\mathsf{H}_{\mathsf{Sign}}$, such that $\mathsf{H}_{\mathsf{Sign}}(\mathsf{pk}, \prod_{j\in\mathcal{S}} R_j \cdot s, m) = h$ and outputs its commitment R_i to the adversary. Since the adversary does not know the random commitment R_i at the time of reprogramming the random oracle $\mathsf{H}_{\mathsf{Sign}}$, the probability that the adversary queried the random oracle $\mathsf{H}_{\mathsf{Sign}}$ on the particular input is negligible. When the adversary provides all shares $\{R_j\}_{j\in\mathcal{S}\setminus i}$ to the reduction, the reduction outputs the previously stored partial signature z_i. Since the reduction reprogrammed the random oracle, this partial signature z_i is indistinguishable from a partial pre-signature w.r.t. the statement s from the adversary's point of view.

With this intuition in mind, we now start the formal proof of Lemma 1. We assume towards contradiction that there exists an adversary $\mathcal{A}$ that breaks the full extractability of TAS and build a reduction $\mathcal{B}$ that breaks the strong unforgeability of Sparkle or the hardness of the relation $\mathcal{R}$. This reduction $\mathcal{B}$ simulates the game Ext perfectly to the adversary, which we show using a series of game hops with negligible transition.

Game $\mathcal{G}_0$: The initial game $\mathcal{G}_0$ is the original Ext game. Since $\mathcal{G}_0$ simulates the Ext game perfectly, it holds that $\Pr[\mathsf{Ext}(\lambda) = 1] = \Pr[\mathcal{G}_0(\lambda) = 1]$.

Game $\mathcal{G}_1$: The first game $\mathcal{G}_1$ differs from game $\mathcal{G}_0$ when the event $\mathsf{Break}_{\mathcal{R}}$ happens. $\mathsf{Break}_{\mathcal{R}}$ occurs if the adversary outputs a forgery, which allows breaking a challenge instance of the hard relation (i.e., a statement that was output by the oracle NewS). If $\mathsf{Break}_{\mathcal{R}}$ happens, the game $\mathcal{G}_1$ aborts. This game hop does not differ from Ext if the adversary returns an adapted pre-signature on a non-challenge statement since the winning condition covers this already.

Claim 1. *Let* $\mathsf{Break}_{\mathcal{R}}$ *be the event, whereby* $\mathcal{G}_1$ *aborts. Then, the adversary* $\mathcal{A}$ *found a signature forgery* σ^* *that allows extracting a challenge witness using a previously output pre-signature. If* $\mathsf{Break}_{\mathcal{R}}$ *happens, then we can build a reduction* $\mathcal{B}'$ *that can break the hardness of the hard relation* $\mathcal{R}$*. Thus,* $\mathsf{Break}_{\mathcal{R}}$ *occurs only with negligible probability* $\mathsf{negl}_1(\lambda)$*.*

Proof of the claim. To show this claim, we build a reduction $\mathcal{B}$ against the hardness of the hard relation $\mathcal{R}$. As input, $\mathcal{B}$ gets a challenge statement Y'. The reduction $\mathcal{B}$ guesses which of the polynomially many requested challenge statements will be broken by $\mathcal{A}$ and replaces this statement with Y'. When the game $\mathcal{G}_1$ aborts as $\mathsf{Break}_{\mathcal{R}}$ happened, $\mathcal{B}$ extracts a witness for s' using the extract algorithm, the signature forgery σ^* and the corresponding pre-signature. This extracting yields a valid witness since $\mathsf{Break}_{\mathcal{R}}$ happened. $\mathcal{B}$ then outputs this valid witness for s'. The probability that

$\mathcal{B}$ is successful is polynomially bounded by the ability of the adversary to cause the event $\mathsf{Break}_{\mathcal{R}}$ to happen. As we assume that the relation $\mathcal{R}$ is a hard relation, such a $\mathcal{B}$ cannot exist, and hence, the gap between the games $\mathcal{G}_0$ and $\mathcal{G}_1$ is negligible. □

GAME $\mathcal{G}_2$: In this game, we simulate the pre-signing and signing oracles using the signing oracles provided by the strong unforgeability game of Sparkle. As input, $\mathcal{B}$ receives the public key pk and forwards pk to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs the corrupted set corr, and $\mathcal{B}$ also outputs corr. Eventually, $\mathcal{B}$ receives the corrupted signing keys and forwards these to $\mathcal{A}$. When $\mathcal{A}$ queries the signing oracle, $\mathcal{B}$ forwards the request to its own signing oracle. When $\mathcal{A}$ queries the pre-signing oracle on a message m and a statement Y, $\mathcal{B}$ randomly samples a message m' and requests its signing oracle on m'. It receives a partial signature of the form $(R_i, \mathsf{sk}_i \cdot h + r_i)$. Then, $\mathcal{B}$ programs the RO, such that $\mathsf{H}_{\mathsf{Sign}}(\mathsf{pk}, R \cdot s, m) = h$. Afterward, $\mathcal{B}$ interacts with the other signers and extracts from the ROM, such that the final random commitment of the random signature equals $Y \cdot R$ as described at the beginning of this proof. The challenger for the unforgeability of sparkle simulates the keys and oracles perfectly, but $\mathcal{A}$ can notice a gap if it queries the random oracle before $\mathcal{B}$ reprograms it accordingly.

Claim 2. *Let* coll *denote the event where* $\mathcal{G}_2$ *aborts due to an unprogrammable pre-signature. In this scenario, the adversary* $\mathcal{A}$ *must have queried the random oracle* H_{cm} *with the honest signer's random commitment* R_i*, or queried the random oracle* $\mathsf{H}_{\mathsf{Sign}}$ *with an input of* $(\mathsf{pk}, R \cdot s, m)$*,* before *obtaining the random share* R_i *from the honest signer. Since the set of statements is exponentially large,* coll *occurs only with negligible probability* $\mathsf{negl}_2(\lambda)$*.*

When winning the Ext game, the adversaries output (m^*, σ^*) is a valid forgery that breaks the existential unforgeability of $\mathsf{Sparkle}$ with overwhelming probability. The pair (m^*, σ^*) is only not a valid forgery against the existential unforgeability of Sparkle if m^* was previously chosen as a random message when our reduction queried its signature oracle. Since the messages were chosen uniformly at random by the reduction, this is only the case with a probability of at most $q/2^{\lambda}$, where q is the number of queried pre-signatures. This probability is negligible in λ since q is polynomially bounded. □

We defer the other lemmas and proofs to Appendix B.1.

5 Certified Witness Encryption

This section introduces a new cryptographic primitive, which we call *Certified witness encryption* (CWE). At a high level, CWE is a form of witness encryption where a valid signature serves as the witness required for decryption. More precisely, in an CWE scheme, a party provides a certificate $(\mathsf{vk}, m) = s$ and receives an encryption key ek. Any ciphertext encrypted under ek can only be decrypted by someone who possesses a valid signature σ on m with respect to vk. This naturally defines a witness encryption scheme for the language

$$\mathcal{L}_{\mathcal{R}} := \{((\mathsf{vk}, m), \sigma) = (s, w) \mid \mathsf{Vrfy}(\mathsf{vk}, \sigma, m) = 1\}.$$

The security of CWE aligns with the standard notion of Chosen Plaintext Attack (CPA) security for encryption schemes, with one key difference: the adversary is allowed to choose the message m

defining the language $\mathcal{L}_{\mathcal{R}}$. Intuitively, CWE should remain CPA-secure even when the adversary selects the certificate's message.

We formalize CWE, define its corresponding security notion, and present a construction based on Schnorr signatures and ElGamal encryption in Section 5. Additionally, we provide a general framework for constructing CPA-secure CWE encryption from a broad class of signature schemes in Appendix D. Our framework applies to signatures based on the DDH and CDH assumptions, including the Camenisch-Lysyanskaya (CL) [CL03], Waters+ [BSW06], and Katz-Wang [KW03] signature schemes.

Definition of Certified Witness Encryption. A CWE scheme is defined w.r.t. some signature scheme $\Pi_\Sigma = (\mathsf{KGen}_\Sigma, \mathsf{Sign}, \mathsf{Vrfy})$ for the relation $\mathcal{L}_{\mathcal{R}}$ as defined above. The key generation algorithm $\mathsf{KG}_{\mathsf{CWE}}$ takes as input the statement s and outputs an encryption key ek only. The encryption algorithm $\mathsf{Enc}_{\mathsf{CWE}}$ encrypts messages under ek and the resulting ciphertexts can be decrypted using a witness w. We emphasize that while statements for $\mathcal{L}_{\mathcal{R}}$ can be efficiently sampled, generating valid statement-witness pairs (s, w) efficiently requires knowledge of a signing key corresponding to the verification key vk.

Definition 15 (Certified Witness Encryption)**.** *A* certified witness encryption *scheme* $\Pi_{\mathsf{CWE}} = (\mathsf{KG}_{\mathsf{CWE}}, \mathsf{Enc}_{\mathsf{CWE}}, \mathsf{Dec}_{\mathsf{CWE}})$ *w.r.t. a signature scheme* Π_{Sign} *and the hard relation* $\mathcal{R}$ *with statement-witness pairs* (s, w) *that corresponds to the language* $\mathcal{L}_{\mathcal{R}}$ *as defined above and consists of the following efficient algorithms:*

$\underline{\mathsf{ek} \leftarrow \mathsf{KG}_{\mathsf{CWE}}(s)}$*: The input of the* key generation *algorithm is a statement* $s \in \mathcal{L}_{\mathcal{R}}$ *and it returns an encryption key* ek*.*

$\underline{ct \leftarrow \mathsf{Enc}_{\mathsf{CWE}}(\mathsf{ek}, m)}$*: The* encryption *algorithm takes as input a public key* ek *and a message* m*, it outputs a ciphertext* c*.*

$\underline{m \leftarrow \mathsf{Dec}_{\mathsf{CWE}}(w, ct)}$*: The input of the* decryption *algorithm is a witness* w *and a ciphertext* c*, it outputs a message* m*.*

Let $\Pi_\Sigma = (\mathsf{KGen}_\Sigma, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme for the relation $\mathcal{R}$ as defined above. A certified witness-encryption scheme $\Pi_{\mathsf{CWE}} = (\mathsf{KG}_{\mathsf{CWE}}, \mathsf{Enc}_{\mathsf{CWE}}, \mathsf{Dec}_{\mathsf{CWE}})$ is *correct*, if for all security parameter λ, all pairs $(s, w) \leftarrow \mathsf{genR}(1^\lambda)$, all $\mathsf{ek} \leftarrow \mathsf{KG}_{\mathsf{CWE}}(s)$, all messages m, we have $m = \mathsf{Dec}_{\mathsf{CWE}}(w, \mathsf{Enc}_{\mathsf{CWE}}(\mathsf{ek}, m))$.

In cases where we need verifiability of our encryption, we extend the functionality with the algorithms $(\mathsf{VEnc}_{\mathsf{CWE}}, \mathsf{Vrfy}_{\mathsf{CWE}})$ similar to verifiable encryption as defined in Appendix A.5. The algorithm $\mathsf{VEnc}_{\mathsf{CWE}}$ takes as input an encryption key ek, a message m, and a commitment M to m. It outputs a verifiable ciphertext ct. The algorithm $\mathsf{Vrfy}_{\mathsf{CWE}}$ takes as input an encryption key ek, a commitment M, and a verifiable ciphertext ct. It outputs a bit b indicating whether c encrypts a valid opening of M with respect to ek. These algorithms can be instantiated by applying simulation-sound NIZK schemes to our encryption protocols.

Security of Certified Witness Encryption. We formalize the security of CWE similar to CPA security for encryption schemes, but we let the adversary choose the message m that is used for the computation of the witness. More precisely, we consider the hard relation $\mathcal{R}(1^\lambda, m)$ with auxiliary information m for the language

$$\mathcal{L}_{\mathcal{R}} := \{((\mathsf{vk}, m), \sigma) = (s, w) \text{ s.t. } \mathsf{Vrfy}(\mathsf{vk}, \sigma, m) = 1\}.$$

Since every CWE is defined with respect to a signature scheme Π_Σ, we give the attacker $\mathcal{A}$ the public key vk of the signature scheme as input. The attacker outputs two equal messages m_0, m_1 and one message m for the key derivation; it receives the encryption c_b of the message m_b under ek as well as the statement s as input and outputs its choice bit b. Our definition is sufficient for our purposes and we leave the notion of security, e.g., where to get access to the signing oracle, open for future research.

Definition 16 (CPA Security of CWE)**.** *Let $\mathcal{R}$ be a hard relation for the language $\mathcal{L}_\mathcal{R}$, and $\Pi_\Sigma = (\mathsf{KGen}_\Sigma, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme. A* certified witness encryption *scheme $\Pi_\mathsf{CWE} = (\mathsf{KG}_\mathsf{CWE}, \mathsf{Enc}_\mathsf{CWE}, \mathsf{Dec}_\mathsf{CWE})$ has* indistinguishable encryptions under a chosen-plaintext attack *if for every PPT adversary $\mathcal{A}$, there exists a negligible function negl such that*

$$\left| \Pr\left[\mathsf{CWE} - \mathsf{CPA}^1_{\mathcal{A},\Pi_\mathsf{CWE}}(\lambda) = 1\right] - \Pr\left[\mathsf{CWE} - \mathsf{CPA}^0_{\mathcal{A},\Pi_\mathsf{CWE}}(\lambda) = 1\right] \right| \leq \mathsf{negl}(\lambda),$$

where the experiment $\mathsf{CWE} - \mathsf{CPA}^b_{\mathcal{A},\Pi_\mathsf{CWE}}$ is described in Figure 8, and the probabilities are taken over the random choices of all probabilistic algorithms.

$\mathsf{CWE} - \mathsf{CPA}^b_{\mathcal{A},\Pi_\mathsf{CWE},\Pi_\Sigma,\mathcal{R}}(\lambda)$

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_\Sigma(\lambda)$
2 : $(m_0, m_1, m) \leftarrow \mathcal{A}(\mathsf{vk})$ // *s.t.* $|m_0| = |m_1|$
3 : $(s, w) \leftarrow \mathsf{genR}(1^\lambda, m)$
4 : $\mathsf{ek} \leftarrow \mathsf{KG}_\mathsf{CWE}(s)$
5 : $c_b \leftarrow \mathsf{Enc}_\mathsf{CWE}(\mathsf{ek}, m_b)$
6 : $b' \leftarrow \mathcal{A}(c_b, s)$
7 : **return** $b' == b$

Figure 8: The security game $\mathsf{CWE} - \mathsf{CPA}^b_{\mathcal{A},\Pi_\mathsf{CWE}}(\lambda)$.

We propose a certified witness encryption scheme for Schnorr signatures. This scheme adapts the ElGamal public-key encryption scheme but uniquely incorporates Schnorr signatures [Sch91] to serve as decryption witnesses.

CWE for Schnorr Signatures. Schnorr signatures are defined within a group $\mathbb{G}$ of prime order p with a generator g and a hash function H. For a given verification key $\mathsf{vk} = g^\mathsf{sk}$ and message m, a Schnorr signature has the form $\sigma = (R, z) = (g^r, \mathsf{sk} \cdot h + r)$, where h is obtained from $\mathsf{H}(\mathsf{vk}, R, m)$ and r is a randomly chosen element of $\mathbb{Z}_p$. Verifying a Schnorr signature involves checking whether $\mathsf{vk}^h \cdot R$ equals g^z.

Given the intrinsic nature of Schnorr signatures, where each message-key pair can generate an exponential number of witnesses (i.e., different signatures corresponding to each element of randomness in $\mathbb{Z}_p$), we design our construction to avoid the need to decrypt for every potential witness. I.e., we formulate a specific hard relation for Schnorr signatures that possesses unique witnesses: The statement of the relation includes the verification key vk, the message m, and a commitment to the randomness R. The witness for such a statement is a Schnorr signature (R, z), incorporating the random commitment R. More formally, we define the hard relation w.r.t. the

language

$$\mathcal{L}_{\mathcal{R}} := \{((\mathsf{vk}, m, R), (R, z)) = (s, w) \text{ s.t. } \mathsf{Vrfy}(\mathsf{vk}, (R, z), m) = 1\}.$$

This relation is as hard as computing a valid signature Schnorr signature on the verification key vk, the message m, and the random commitment R. In the next step, we build a CWE scheme for this hard relation.

To compute encryption keys, we observe the structure of the verification algorithm of Schnorr signatures: Having available the values (vk, m, R), one can locally compute the value $\mathsf{vk}^h \cdot R$, which equals g^z. The encryption of a message m_{Enc} using the encryption key ek follows a process similar to ElGamal encryption: We sample a random element $y \leftarrow\!\!\$\ \mathbb{Z}_p$ and output the pair $(g^y, \mathsf{ek}^y \cdot m)$. To decrypt a ciphertext (c_1, c_2) using a signature (R, z), we compute $m = c_2/(c_1^z)$. More formally, we construct CWE for Schnorr signatures in Construction 2.

Construction 2 (DDH-Based CWE encryption scheme)**.** *Let $\mathbb{G}$ be a group of prime order p with a generator g. We define the CWE scheme $\Pi_{\mathsf{CWE}} = (\mathsf{KG}_{\mathsf{CWE}}, \mathsf{Enc}_{\mathsf{CWE}}, \mathsf{Dec}_{\mathsf{CWE}})$ for Schnorr signatures w.r.t. $(\mathbb{G}, p, g)$ in Figure 9.*

$\mathsf{KG}_{\mathsf{CWE}}(\mathsf{vk}, m, R)$	$\mathsf{Enc}_{\mathsf{CWE}}(\mathsf{ek}, m)$	$\mathsf{Dec}_{\mathsf{CWE}}(\sigma, c)$
$h \leftarrow \mathsf{H}(\mathsf{vk}\|\|R\|\|m)$	$y \leftarrow\!\!\$\ \mathbb{Z}_p$	parse c as (c_1, c_2)
$\mathsf{ek} := \mathsf{vk}^h \cdot R$	**return** $(g^y, \mathsf{ek}^y \cdot m)$	parse σ as (R, z)
return ek		**return** $c_2/(c_1^z)$

Figure 9: Schnorr-based CWE encryption from DDH. In Appendix D, we extend this construction into a "signed" setting by leveraging the random oracle (RO) to mask encrypted messages. The use of the RO enables a security proof based on the Computational Diffie-Hellman (CDH) assumption.

This construction is correct and CPA secure. We show correctness in Lemma 9 in the appendix and CPA security in Theorem 2.

Theorem 2 (Security of Schnorr CWE)**.** *If the decisional Diffie-Hellman (DDH) assumption is hard, and H is a compressing, non-zero hash function, then Construction 2 is a CPA secure CWE scheme.*

Proof Sketch. By contradiction, we assume that there exists an adversary $\mathcal{A}$ that breaks the CPA security of Π_{CWE} with non-negligible probability. We use this adversary to build a reduction from the CPA security of Π_{CWE} to the hardness of DDH. The reduction $\mathcal{R}$ gets as input the values $([\alpha], [\beta], [\gamma])$, such that γ is either a random value or $\alpha \cdot \beta$. The reduction forwards $[\alpha]$ as verification key to $\mathcal{A}$, which eventually outputs three messages (m_i, m_1, m). $\mathcal{R}$ samples a fresh randomness, commitment pair $(R, r) \leftarrow \mathsf{genR}(1^\lambda)$ and computes the challenge ciphertext on the challenge message m_b via $c_b \leftarrow ([\beta], [\beta]^r \cdot [\gamma]^h \cdot m_b)$. The reduction forwards c_b to $\mathcal{A}$, which eventually outputs a bit b' and $\mathcal{R}$ returns $b' == b$. If $\gamma = \alpha \cdot \beta$, then the ciphertext is exactly constructed as the adversary expects it to be

$$c := ([\beta], [\beta]^r \cdot [\gamma]^h \cdot m_b) = ([\beta], [\beta \cdot r + \alpha \cdot \beta \cdot h] \cdot m_b) = ([\beta], [\beta \cdot (r + \alpha \cdot h)] \cdot m_b).$$

If $\gamma \neq \alpha \cdot \beta$, then γ the challenge ciphertext c_b is also a random element in $\mathbb{Z}_p$ (as H is a non-zero function) and is independent of the challenge message m_b. We show in the full proof (Appendix B.3), that if $\gamma \neq \alpha \cdot \beta$, then $\mathcal{A}$ wins with probability at most $1/2$, and if $\gamma = \alpha \cdot \beta$, then $\mathcal{A}$ wins with non-negligible advantage $1/2 + \varepsilon$. Therefore, it holds that:

$$\begin{aligned}&\left|\Pr\left[\mathsf{DDH}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{DDH}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \\ &\left|\Pr\left[\mathsf{CPA}^1_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{CPA}^0_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \left|\frac{1}{2} + \varepsilon - \frac{1}{2}\right| \geq \varepsilon.\end{aligned}$$

This contradicts the hardness of DDH; thus, no efficient adversary against the CPA security of Construction 2 can exist. □

6 A Fair-Exchange Protocol

In this section, we provide a construction for a Schnorr-based MPFE and formally prove its security. Our Multi-Party Fair Exchange MPFE framework uses threshold adaptor signatures as a fundamental component. We split our protocol into three phases: The commit phase that is executed amongst the buyers, the settle phase that is executed amongst the sellers and the reveal phase that is run by each buyer locally. In the commit phase, all buyers jointly compute a pre-signature as a public state. They forward this pre-signature to the sellers, who run the settle protocol to jointly adapt the pre-signature and post it alongside the payment transaction to the blockchain. In the reveal phase, each buyer obtains the latest blockchain state and locally extracts the witness w. A notable challenge using this approach is the lack of fairness during both the pre-signing and the adapting phase, which we address in the commit and settle protocols.

Construction 1 (Schnorr-Based MPFE)**.** *Let $\mathcal{R}$ be a relation for the DLog language, and* TAS *be the threshold adaptor signature based on sparkle. Let* CWE *be a verifiable certified witness encryption based on Schnorr signatures. We define the protocols* Setup, Commit, Eval, Reveal *in Figures 15, 10, and 11.*

Setup. The setup algorithm generates the private and public keys for each protocol participant. Each buyer receives a threshold signing key $\mathsf{sk}_{\mathsf{TS},i}$ shared amongst the buyers and a random key k, which is consistent amongst all buyers. The two keys are combined into the key $\mathsf{sk}_{\mathsf{B}_i} = (\mathsf{sk}_{\mathsf{TS},i}, k)$. Each seller receives a public-secret key pair for encryption $\mathsf{sk}_{\mathsf{S}_j} = \mathsf{sk}_{\mathsf{Enc},j}$, which are combined into the seller secret key $(\mathsf{sk}_{\mathsf{TS},j}, \mathsf{sk}_{\mathsf{Enc},j})$. In addition, the setup algorithm outputs a shared public signing key for all buyers and one per seller. From this point forward, we assume that all public keys generated by setup are known to all protocol participants, and we abuse the notation of $\mathsf{pk}_{\mathsf{S}_j}$, which refers to the public encryption key when encrypting elements and the public signature verification key when verifying signatures.

Commit. The commit protocol enables each buyer to establish an internal state state, which ensures that a witness can be revealed once a designated signature corresponding to a valid payment appears on the blockchain. Additionally, the protocol generates an external state ρ, which is forwarded to the sellers. In our construction, ρ consists of a pre-signature computed with respect to pk_{B} and a rerandomization factor that blinds the final witness from the sellers, preserving witness privacy. Each buyer begins by computing a randomization factor for every seller as $e_j \leftarrow \mathsf{H}(k, \mathsf{tx}, s, j)$,

where k is the buyer-specific secret key and j denotes the seller's identifier. These individual shares are then combined using Lagrange interpolation over a degree-t_{S} polynomial to derive a shared rerandomization factor e. This factor can be reconstructed with knowledge of at least t_{S} shares or, alternatively, by any buyer possessing the key k. The rerandomization factor e is fundamental for ensuring witness privacy. Since we assume that at most $t_{\mathsf{S}} - 1$ sellers may be corrupted, no adversarial seller can learn e, ensuring that the witness remains perfectly blinded. When a buyer later extracts the blinded witness $w + e$, knowledge of k allows them to recover e and ultimately obtain w in plaintext. To commit to a transaction, buyers must first agree on a pre-signature randomness $\widetilde{R}$ using the threshold adaptor signature algorithms pSign^0 and pSign^1. Once consensus on $\widetilde{R}$ is reached, they jointly lock the input transaction on-chain for a sufficient duration. This locking mechanism ensures that the transaction remains accessible to buyers after a predefined time while allowing the adapted pre-signature to spend the transaction beforehand. To maintain clarity, we defer the full details of this locking procedure to the seller-fairness proof. Once the transaction is locked, each buyer locally executes pSign^2 to generate a partial pre-signature ρ'', which is then verifiably encrypted using the CWE scheme. The use of CWE ensures that, once the adapted pre-signature is posted on the blockchain, all buyers can decrypt the ciphertexts and reconstruct the pre-signature—even in cases where some buyers attempt to abort the protocol unfairly. Before proceeding, each buyer verifies the encrypted shares from all other buyers. After successful verification, they send their respective partial pre-signatures ρ'' to the group. Upon collecting a sufficient number of pre-signature shares, each buyer finalizes their internal state state, storing all ciphertexts related to the pre-signature shares. Simultaneously, they output the pre-signature and rerandomization factors as an external state ρ. A formal definition of our Commit construction is provided in Figure 10.

Settle. The settlement protocol ensures that sellers collaboratively adapt the pre-signature into a valid signature and submit it to the blockchain alongside the corresponding transaction. The process unfolds as follows: Each seller sends their blinded witness $w_j + e_j$ to all other sellers. Once a seller collects at least t_{S} blinded witnesses, they compute the final blinded witness $w + e$ and use it to adapt the pre-signature into a valid signature. The adapted pre-signature is then posted on-chain, completing the exchange. From a security perspective, this mechanism ensures that a valid payment is triggered once the adapted pre-signature appears on the blockchain. Since the adversary has locked the transaction inputs to require this pre-signature, and an honest majority of participating sellers provide their blinded witness shares in time, the protocol guarantees that a valid signature will eventually be posted. We formally define the settle protocol in Figure 11.

Reveal. The reveal algorithm enables each buyer to extract the witness w using the adapted signature σ and their internal state. This internal state consists of verifiable certified witness encryption (CWE) ciphertexts containing the pre-signature shares of other buyers. Once the adapted signature σ is posted on the blockchain, each buyer decrypts the CWE ciphertexts to recover the pre-signature shares. Looking ahead, the security analysis will show that if the sellers manage to post a valid signature on the blockchain, each buyer has a sufficient number of pre-signature shares in its local state to reconstruct the pre-signature. After reconstructing the pre-signature, the buyer runs the extraction algorithm from the threshold adaptor signature scheme to obtain the blinded witness $w + e$. Since the buyer knows the blinding factor e, they can efficiently recover the witness w. We give a formal description of the reveal algorithm in Figure 11.

$\mathsf{B}_i.\mathsf{Commit}(\mathsf{sk}_{\mathsf{B}_i}, s, \mathsf{tx})$

1 : // Compute rerandomization factors.
2 : $e_j \leftarrow \mathsf{H}(k, \mathsf{tx}, s, j); e = \sum e_j \lambda_j; E \leftarrow g^e$

3 : // Agree on a pre-signature randomness $\widetilde{R}$
4 : $(\rho_i, \mathsf{st}_i) \leftarrow \mathsf{pSign}^0(i, \mathsf{tx}, s \cdot E, \mathcal{B})$
5 : send ρ_i to all B_j; upon receiving all other ρ_i
6 : $(\rho'_i, \mathsf{st}_i) \leftarrow \mathsf{pSign}^1(\mathsf{st}_i, \{\rho_i\}_{i\in\mathcal{B}}, \mathcal{B})$
7 : send ρ'_i to all B_j; upon receiving all other ρ'_i
8 : $\widetilde{R} \leftarrow \prod_{j\in\mathcal{B}} \rho'_j \cdot E \cdot s$

9 : // Lock the input transaction
10 : $\mathcal{C}.\mathsf{lock}(\mathsf{sk}_{\mathsf{B}_i}, \mathsf{tx}, \widetilde{R})$

11 : // Encrypt the pre-signature share using the CWE scheme
12 : $\mathsf{ek} \leftarrow \mathsf{KG}_{\mathsf{CWE}}(\mathsf{pk}, \mathsf{tx}, \widetilde{R}, \mathsf{tx})$
13 : $\rho''_i \leftarrow \mathsf{pSign}^2(\mathsf{st}_i, \mathsf{sk}_{\mathsf{B}_i}, \{\rho'_i\}_{i\in\mathcal{B}}, \mathcal{B})$
14 : $com_i \leftarrow \mathsf{VEnc}_{\mathsf{CWE}}(\mathsf{ek}, \rho''_i, \widetilde{R})$
15 : send com_i to all B_j

16 : // Reveal the pre-signature share after verifying all commitments
17 : upon receiving all other com_j :
18 : **assert** $\mathsf{Vrfy}_{\mathsf{CWE}}(com_j, \mathsf{pk}_{\mathsf{B}_j}, \widetilde{R}, \mathsf{tx})$
19 : send ρ''_i to all B_j; upon receiving all other ρ''_j :
20 : $\widetilde{\sigma} \leftarrow \mathsf{Combine}(\{\rho'_i, \rho''_i\}_{i\in\mathcal{B}})$
21 : $\mathsf{state}_i \leftarrow (\{com_i\}_{i\in\mathcal{B}}, e); \rho \leftarrow (\widetilde{\sigma}, \mathsf{tx}, \{\Pi_{\mathsf{Enc}}.\mathsf{Enc}(\mathsf{pk}_{\mathsf{S}_j}, e_j)\}_{1\le j\le n_{\mathsf{S}}})$
22 : **return** (state_i, ρ)

Figure 10: Construction of the Commit protocol.

6.1 Security

We now establish the security of our MPFE construction. Initially, our goal was to analyze the security of the MPFE scheme in relation to the CPA security of the CWE scheme. However, in our construction, a partial pre-signature is encrypted using the CWE scheme, and the adapted and combined pre-signature effectively functions as the decryption key. As a result, a direct reduction to the CPA security of CWE is not possible. Instead, we adopt the framework of transparent reductions [Ger+24a] and prove the security of our MPFE scheme based on the hardness of the CDH assumption in the underlying group. Additionally, we ensure that the random oracle is

$\mathsf{S}_i.\mathsf{Settle}^{\mathcal{C}}(\mathsf{sk}_{\mathsf{S}_i}, \rho, w_i)$

1 : parse ρ as $(\widetilde{\sigma}, \mathsf{tx}, \{ct_j\}_{1 \le j \le n_\mathsf{S}})$
2 : $e_i \leftarrow \Pi_{\mathsf{Enc}}.\mathsf{Dec}(\mathsf{sk}_{\mathsf{S}_i}, ct_i), w_i' = w_i + e_i$
3 : send w_i' to all Sellers
4 : upon receiving all other w_j'
5 : $w' \leftarrow \sum_{j \in \mathcal{S}} w_j'$
6 : $\sigma \leftarrow \mathsf{Adapt}(\mathsf{pk}_\mathsf{B}, \widetilde{\sigma}, w')$
7 : $\mathcal{C}.\mathsf{post}(\mathsf{tx}, \sigma)$
8 : **return**

$\mathsf{Reveal}.\mathsf{B}_i(\mathsf{pk}_\mathsf{B}, \mathsf{state}_i, \sigma)$

1 : parse state_i as $(\{com_j\}_{j \in \mathcal{B}}, e)$
2 : parse σ as (R, z)
3 : $\mathsf{dk} \leftarrow z$
4 : **foreach** $j \in \mathcal{B}$ **do** :
5 : $\quad \rho'' \leftarrow \mathsf{Dec}_{\mathsf{CWE}}(\mathsf{dk}, com_j)$
6 : $\widetilde{\sigma} \leftarrow \mathsf{Combine}(\mathsf{pk}_\mathsf{B}, \{\rho''\}_{j \in \mathcal{B}})$
7 : $w' \leftarrow \mathsf{Extract}(\mathsf{pk}_\mathsf{B}, \widetilde{\sigma}, \sigma)$
8 : **return** $w' - e$

Figure 11: The Settle Protocol and the reveal algorithm.

programmed in a way that prevents circular security.

Security Parameters. We prove the security of our MPFE scheme under optimal bounds. Specifically, we establish fairness guarantees based on threshold conditions: buyer fairness holds for any $t_\mathsf{B} < n_\mathsf{B}$ in the presence of a minimal blockchain, while seller fairness requires t_S honest *participating* sellers. (i.e., $n_\mathsf{S} \ge 2 \cdot t_\mathsf{S} - 1$) in the presence of a minimal blockchain with timing. Our impossibility result confirms that these conditions are both necessary and sufficient. For our security analysis, we assume that at least t_S honest sellers participate in the exchange. This assumption is practical, as sellers can enforce agreement through a voting mechanism before executing the protocol and enforce the participation of at least $2 \cdot t_\mathsf{S} - 1$ sellers. To keep our construction concise, we leave the voting process out of scope and focus solely on the cryptographic core of the protocol. Honest sellers only participate if the required threshold is met. To faciliate notations, we talk about t_S-seller fairness (t_B-buyer fairness) and assume, that in these cases $t_\mathsf{S} - 1$ sellers ($t_\mathsf{B} - 1$ buyers) are corrupted.

Theorem 3. *Let $\mathcal{R}$ be a hard relation for the DLog language in the group $\mathbb{G}$, and let* TAS *be the secure Schnorr threshold adaptor signature scheme defined in Construction 1. Let* CWE *be a verifiable certified witness encryption scheme for Schnorr signatures, as defined in Construction 4, using a simulation-sound non-interactive zero-knowledge proof of knowledge* (NIZK). *Let Π_{Enc} be a CPA secure public key encryption scheme. If the CDH assumption holds in $\mathbb{G}$, then Construction 1 is a secure* MPFE *protocol in the random oracle model. Specifically, Construction 1 guarantees:*

- *t_B-buyer fairness for any $t_\mathsf{B} \le n_\mathsf{B}$ in the presence of a minimal blockchain;*
- *t_S-seller fairness for t_S participating honest sellers (i.e., $n_\mathsf{S} \ge 2 \cdot t_\mathsf{S} - 1$) in the presence of a minimal blockchain with timing;*
- *witness privacy for any $t_\mathsf{S} \le n_\mathsf{S}$.*

To prove Theorem 3, we show buyer fairness, seller fairness, and witness privacy in separate lemmas. We show buyer fairness now and defer the other lemmas and proofs to Appendix B.2.

Lemma 2 (Buyer Fairness)**.** *Let $\mathcal{R}$ be a hard relation for the DLog language in the group $\mathbb{G}$, and let* TAS *be the secure Schnorr threshold adaptor signature scheme defined in Construction 1. Let* CWE *be a verifiable certified witness encryption scheme for Schnorr signatures, as defined in Construction 4, using a simulation-sound non-interactive zero-knowledge proof of knowledge. If the CDH assumption holds in $\mathbb{G}$, then Construction 1 achieves t_{B}-buyer fairness for any $t_{\mathsf{B}} \leq n_{\mathsf{B}}$.*

Proof of Lemma 2. To prove Lemma 2, we construct a reduction from the buyer fairness of our construction to one of the following security properties:

- The *simulation soundness* of the NIZK.
- The *hardness of the Computational Diffie-Hellman (CDH)* problem in $\mathbb{G}$.
- The *extractability* of the threshold adaptor signature scheme.

We assume, for contradiction, that there exists an efficient adversary $\mathcal{A}$ that breaks the buyer fairness of Construction 1. Since the `settle` and `reveal` algorithms are executed locally by each party, the adversary can only violate buyer fairness by interacting with honest parties in the commit protocol. To succeed, the adversary must output a valid signature signed by the buyers while ensuring that at least one buyer cannot reconstruct the corresponding challenge witness. In our reduction, the interaction with honest buyers is simulated by providing buyer oracles to the adversary. We define two events based on the type of forgery the adversary produces:

- **Fresh forgery** (fresh): The forgery is fresh if it corresponds to a random commitment $\widetilde{R}$ and message m that was never used in a signing session when interacting with the oracles.
- **Non-fresh forgery** ($\neg\mathsf{fresh}$): The forgery is non-fresh if the pair $(\widetilde{R}, m)$ was previously used in a signing session.

The adversary wins if either of these cases holds and it holds that

$$\Pr[\mathsf{BF}(\lambda) = 1] = \Pr[\mathsf{BF}(\lambda) = 1 \wedge \mathsf{fresh}] + \Pr[\mathsf{BF}(\lambda) = 1 \wedge \neg\mathsf{fresh}].$$

We analyze both cases independently and begin by analyzing the **non-fresh** case: In the non-fresh case, we distinguish two subcases. The first case occurs when the adversary $\mathcal{A}$ aborts the commit protocol before receiving the plaintext pre-signature shares (Line 19 in Figure 10). We define the event where the adversary does not wait for the shares as EA. Looking ahead, we will reduce the case EA to the hardness of CDH. The second case occurs when $\mathcal{A}$ completes the interaction and receives the pre-signature shares before forging a signature. We will reduce this case to the simulation soundness of the NIZK. It holds, that

$$\Pr[\mathsf{BF}(\lambda) = 1 \wedge \mathsf{fresh}] = \Pr[\mathsf{BF}(\lambda) = 1 \wedge \mathsf{fresh} \wedge \mathsf{EA}] + \Pr[\mathsf{BF}(\lambda) = 1 \wedge \mathsf{fresh} \wedge \neg\mathsf{EA}].$$

Since the event EA is a part of the event fresh, we omit fresh for better readability. To show that fresh forgeries happen only with negligible probability, we analyze these two subcases separately and show that both probabilities are negligible. In the first case, EA, the adversary outputs a valid signature before receiving its commitments to the honest buyer's pre-signature shares. We now construct a reduction that uses this adversary to break the CDH assumption in $\mathbb{G}$. To achieve this,

we perform a sequence of game hops, gradually transforming the adversary's attack into a CDH solver. We begin with the initial game $\mathcal{G}_0$, which is simply the BF security game conditioned on the event EA. By definition, the adversary's success probability in this game equals

$$\Pr[\mathsf{BF}(\lambda) = 1 \wedge \mathsf{EA}] = \Pr[\mathcal{G}_0(\lambda) = 1].$$

In the first game hop, we move to game $\mathcal{G}_1$, where we modify the protocol by replacing the NIZK proofs used in the verifiable CWE encryption with simulated proofs from the simulator of the simulation-sound NIZK scheme. Since the simulation soundness of NIZK ensures that this change does not affect the adversary's ability to distinguish the real and simulated proofs, this transition introduces only a negligible difference in the adversary's success probability:

$$\Pr[\mathcal{G}_0(\lambda) = 1] \leq \Pr[\mathcal{G}_1(\lambda) = 1] + \mathsf{negl}_1(\lambda).$$

Next, in game $\mathcal{G}_2$, we set up a reduction to the hardness of the CDH problem. The reduction receives as input two group elements $[\alpha]$ and $[\beta]$ and must compute the CDH solution $[\gamma] = [\alpha \cdot \beta]$. To do so, the reduction simulates the protocol execution while embedding the CDH challenge into the adversary's environment. The reduction begins by generating random signing key shares sk_i for all parties and forwards the signing keys of the corrupted parties to the adversary. It then guesses the signing session in which the adversary will attempt to break buyer fairness (the guess is efficient, as there are q_{sess} signing sessions which are polynomially bounded). Since we are in the non-fresh case, such a session exists. To embed the CDH challenge, the reduction programs the signature randomness $\widetilde{R}$ in the guessed session to be $[\alpha]$. This is achieved by setting the randomness of a single honest party to

$$R_i = \frac{[\alpha]}{E \cdot s \cdot R'},$$

where R' is the product of the random nonces of all other parties, extracted from the commitments of the first signing round by using the random oracle. To further embed the challenge, the reduction sets the ciphertext com_i for the guessed honest party to

$$com_i = ([\beta], r), \quad \text{where } r \leftarrow\!\!\$\ \mathbb{Z}_p.$$

For all other signing sessions, the reduction answers queries honestly. This is feasible as it knows the signing key shares of all honest parties. The final step in the reduction is extracting the CDH solution. The reduction guesses if and when the adversary queries the random oracle on the CDH solution. This introduces a loss factor of $1/2$ and $1/q_{\mathsf{ro}}$, where q_{ro} is the number of random oracle queries (also polynomially bounded). If the guess is correct and the adversary indeed queries the RO on a solution for CDH, the reduction can compute

$$[\gamma] = [\alpha]^{-h \cdot \mathsf{sk}} \cdot x,$$

where x is the adversary's random oracle query and $h = \mathsf{H}(\mathsf{pk}, \widetilde{R}, m)$. If the guess is correct, the reduction obtains $[\gamma]$, thereby solving the CDH problem. The reduction finishes and the adversary cannot distinguish the simulated com_i from an honest one, as it never receives the RO evaluation on its input. If the adversary does not query the random oracle on a CDH solution, it must still produce a non-fresh forgery. Since the reduction controls the signing key, it can extract $\beta = \widetilde{r}$ from the forgery $(\widetilde{R}, \mathsf{sk} \cdot \mathsf{H}(\mathsf{pk}, \widetilde{R}, m) + \widetilde{r})$, thereby still solving CDH. Introduced by the guessing of the

reduction, obtaining the CDH solution adds a polynomial loss to the reduction. If all guesses are correct, the reduction $\mathcal{B}$ solves the CDH instance. This imposes

$$\Pr[\mathcal{B}([\alpha],[\beta]) = [\alpha \cdot \beta]] = \Pr[\mathcal{G}_2(\lambda,[\alpha],[\beta]) = 1] = \frac{q_{\mathsf{ro}} + 1}{2 \cdot q_{\mathsf{sess}} \cdot q_{\mathsf{ro}}} \cdot \Pr[\mathcal{G}_1(\lambda) = 1].$$

Since we assume the probability of solving CDH to be negligible, and we have a negligible transition from game $\mathcal{G}_1$ to the game $\mathsf{BF} \wedge \mathsf{EA}$, this concludes the proof for the EA case. Specifically, we showed that the adversary's success probability in the case EA is negligible.

We now analyze the second case, where the adversary does not abort early ($\neg\mathsf{EA}$). In this scenario, the adversary successfully completes the interaction, receives the plaintext pre-signature shares, and subsequently produces a non-fresh forgery. This implies that the challenger has verified the NIZK corresponding to the VPKE of the pre-signature shares provided by the adversary, yet at least one honest buyer is unable to reconstruct a valid pre-signature after the adversary outputs the forgery. By the correctness of the CWE scheme and the bijectivity of the Schnorr signature scheme, the challenger is able to decrypt all well-formed commitments com_i that the adversary submitted before the challenger outputs the pre-signature forgery. Since the adversary violates buyer fairness, at least one honest buyer must be unable to reconstruct a valid pre-signature using the decrypted commitments. This violates the well-formedness of the commitments and leads to a reduction against the simulation soundness of the NIZK system. To construct this reduction, the challenger is given a common reference string crs for the NIZK. The challenger runs the setup algorithm for the MPFE scheme, ensuring that it can answer all adversary queries. When the adversary produces a ciphertext that verifies correctly but fails to decrypt into a valid pre-signature share, the challenger outputs the pre-signature encryption along with the corresponding NIZK proof. This contradicts the simulation soundness of the NIZK, demonstrating that the adversary's success probability in this case is negligible. Having established that the adversary's success probability is negligible in both cases—whether it aborts early (EA) or completes the interaction ($\neg\mathsf{EA}$)—we conclude that no efficient adversary can break buyer fairness by outputting a non-fresh signature in Construction 1.

In the case where the forgery is fresh (fresh), the adversary's forgery was not obtained by adapting any pre-signature that had been partially output by an honest buyer. This directly contradicts the extractability of the threshold adaptor signature scheme. This can be shown by a trivial reduction to the extractability: The reduction forwards corrupted keys and answers each oracle query using the provided oracles. Since the forgery is fresh and valid, by the bijectivity of the Schnorr verification, no previously output pre-signature allows for extracting a valid challenge witness. Thus, the reduction breaks extractability whenever the adversary breaks buyer fairness in the fresh case. Thus, the probability of winning with a fresh forgery is negligible.

Summarizing our results, we have divided the adversary's winning probability into three distinct cases: the event where the adversary aborts early and the forgery is non-fresh ($\mathsf{EA} \wedge \neg\mathsf{fresh}$), the event where the adversary completes the interaction but still produces a non-fresh forgery ($\neg\mathsf{EA} \wedge \neg\mathsf{fresh}$), and the event where the forgery is fresh (fresh). We have shown that in each of these cases, the adversary's success probability is negligible. Furthermore, we need to guess which case applies. This guessing introduces, at most, a polynomial loss. Therefore, we conclude that no efficient adversary can successfully violate the buyer fairness of Construction 1. □

7 Impossibility Result

We prove the impossibility of a multiparty fair exchange protocol when using a minimal blockchain, which we model as a public bulletin board with ordering. Our proof follows a two-step approach: First, we show that seller fairness cannot be guaranteed under any threshold assumptions when relying solely on a minimal blockchain. Second, we extend this result by proving that even with the addition of timing mechanisms to the blockchain, fairness remains unattainable in the presence of less than t_{S} honest participating sellers. A MPFE protocol for exchanging a coin for a witness using a minimal blockchain operates in the following setting:

- The witness w is secret-shared among the sellers, requiring at least t_{S} sellers to reconstruct.
- The buyers collectively hold a signing key.
- The blockchain functions as an append-only ordered bulletin board (c.f. Definition 3).
- Each participant maintains an internal state: initially, the buyers hold the signing key, while each seller possesses a share of the witness.

Communication Model. We model communication and computation in a sequential setting. Looking ahead to our adversarial model, adopting a sequential model is reasonable since we assume that the adversary can delay messages on the network. Consequently, we consider messages posted to the blockchain and exchanged between users to be sequential [FY92; Kat24].

Adversarial Model. We assume an adversary who is able to delay every message sent over the network and to the blockchain. In addition, the adversary controls the order in which parties are activated to perform computations. To constrain the adversary, we impose an ordering constraint: no party can compute twice until all other parties have computed at least once in the meantime. While this constraint limits the adversary's power, it ultimately strengthens our impossibility result, since we show that no protocol can exist even under this weakened adversarial model.

Protocol Execution. Using this adversarial model, we structure protocol execution in rounds: At the beginning of each round, the adversary chooses a party to execute in that round. The selected party then receives any pending messages addressed to it that the adversary may have delayed from previous rounds. In addition, the party learns the current state of the blockchain as provided by the adversary. However, the adversary cannot arbitrarily change the state of the blockchain; it can only reorder transactions within the constraints of the protocol and delay their inclusion. After receiving its inputs, the party has polynomial time to perform its computations. At the end of the round, the party forwards any outgoing messages to the adversary, who controls their delivery, and outputs an updated blockchain state. This process continues in successive rounds until all parties have completed protocol execution.

Impossibility. We now show that in our modeled setup, no MPFE protocol that achieves t_{B}-buyer fairness can also achieve t_{S}-seller fairness in the presence of a minimal blockchain.

Theorem 4 (Impossibility of Fairness for Minimal Blockchains)**.** *No multi-party fair exchange protocol Π_{MPFE} that achieves t_{B}-buyer fairness can also achieve t_{S}-seller fairness in the presence of only a minimal blockchain.*

Proof Intuition. We provide an intuition over our proof. If a buyer's payment appears on-chain, the buyer must be able to extract the witness w; otherwise, an adversary could make the buyer pay without revealing w, breaking buyer fairness (Lemma 15). There must exist a critical *last message* (Lemma 16) that enables the final signature σ_{B}, and this message is received by a seller first. Without it, a malicious buyer could extract w without paying, violating seller fairness. The adversary corrupts all buyers and a set of sellers. If a corrupted seller receives the last message (this happens with probability $t_{\mathsf{S}} - 1/n_{\mathsf{S}}$), the adversary learns σ_{B} and extracts w. Before posting σ_{B}, the adversary can invalidate all buyer transactions, ensuring no honest seller is paid. Since w is extracted without full payment, seller fairness is violated. This attack exploits classical *front-running*, where an adversary anticipates and manipulates transaction order to its advantage. Since this attack succeeds with non-negligible probability, achieving fairness for both buyers and sellers is impossible with a minimal blockchain. We defer the full proof to Appendix E. □

Theorem 5 (Impossibility of Fairness with a Majority of Dishonest Sellers)**.** *No multi-party fair exchange protocol Π_{MPFE} that achieves t_{B}-buyer fairness can also achieve t_{S}-seller fairness in the presence of a minimal blockchain with timelocks if not at least t_{S} honest sellers participate in the protocol.*

Proof intuition. If not at least t_{S} honest sellers participate, then for any MPFE protocol, an adversary can ensure that only corrupted parties receive a final message (Lemma 17). This allows the adversary to compute the final signature and learn the witness while no honest seller learns the witness (due to the missing last message for honest sellers). For seller fairness to hold, honest sellers must still receive payments. This could happen either before or after the adversary receives the final message. We show that both cases occur with negligible probability. Thus, the adversary extracts the witness while ensuring at least one honest seller remains unpaid, breaking seller fairness. Since this occurs with non-negligible probability, achieving fairness for both sides is impossible under these conditions. We defer the full proof to Appendix E. □

References

[And+13] Marcin Andrychowicz et al. *Secure Multiparty Computations on Bitcoin*. Cryptology ePrint Archive, Report 2013/784. 2013. URL: https://eprint.iacr.org/2013/784.

[And+14] Marcin Andrychowicz et al. "Fair Two-Party Computations via Bitcoin Deposits". In: *FC 2014 Workshops*. Ed. by Rainer Böhme et al. Vol. 8438. LNCS. Springer, Berlin, Heidelberg, Mar. 2014, pp. 105–121. DOI: 10.1007/978-3-662-44774-1_8.

[Aum+21] Lukas Aumayr et al. "Generalized Channels from Limited Blockchain Scripts and Adaptor Signatures". In: *ASIACRYPT 2021, Part II*. Ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13091. LNCS. Springer, Cham, Dec. 2021, pp. 635–664. DOI: 10.1007/978-3-030-92075-3_22.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. "Short Group Signatures". In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Berlin, Heidelberg, Aug. 2004, pp. 41–55. DOI: 10.1007/978-3-540-28628-8_3.

[BDD20] Carsten Baum, Bernardo David, and Rafael Dowsley. "Insured MPC: Efficient Secure Computation with Financial Penalties". In: *FC 2020*. Ed. by Joseph Bonneau and Nadia Heninger. Vol. 12059. LNCS. Springer, Cham, Feb. 2020, pp. 404–420. DOI: 10.1007/978-3-030-51280-4_22.

[BK14] Iddo Bentov and Ranjit Kumaresan. "How to Use Bitcoin to Design Fair Protocols". In: *CRYPTO 2014, Part II*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8617. LNCS. Springer, Berlin, Heidelberg, Aug. 2014, pp. 421–439. DOI: 10.1007/978-3-662-44381-1_24.

[BSW06] Dan Boneh, Emily Shen, and Brent Waters. "Strongly Unforgeable Signatures Based on Computational Diffie-Hellman". In: *PKC 2006*. Ed. by Moti Yung et al. Vol. 3958. LNCS. Springer, Berlin, Heidelberg, Apr. 2006, pp. 229–240. DOI: 10.1007/11745853_15.

[CD00] Jan Camenisch and Ivan Damgård. "Verifiable Encryption, Group Encryption, and Their Applications to Separable Group Signatures and Signature Sharing Schemes". In: *ASIACRYPT 2000*. Ed. by Tatsuaki Okamoto. Vol. 1976. LNCS. Springer, Berlin, Heidelberg, Dec. 2000, pp. 331–345. DOI: 10.1007/3-540-44448-3_25.

[Cho+17] Arka Rai Choudhuri et al. "Fairness in an Unfair World: Fair Multiparty Computation from Public Bulletin Boards". In: *ACM CCS 2017*. Ed. by Bhavani M. Thuraisingham et al. ACM Press, 2017, pp. 719–728. DOI: 10.1145/3133956.3134092.

[Chu+23] Hien Chu et al. "Practical Schnorr Threshold Signatures Without the Algebraic Group Model". In: *CRYPTO 2023, Part I*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14081. LNCS. Springer, Cham, Aug. 2023, pp. 743–773. DOI: 10.1007/978-3-031-38557-5_24.

[CKM23] Elizabeth C. Crites, Chelsea Komlo, and Mary Maller. "Fully Adaptive Schnorr Threshold Signatures". In: *CRYPTO 2023, Part I*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14081. LNCS. Springer, Cham, Aug. 2023, pp. 678–709. DOI: 10.1007/978-3-031-38557-5_22.

[CL03] Jan Camenisch and Anna Lysyanskaya. "A Signature Scheme with Efficient Protocols". In: *SCN 02*. Ed. by Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano. Vol. 2576. LNCS. Springer, Berlin, Heidelberg, Sept. 2003, pp. 268–289. DOI: 10.1007/3-540-36413-7_20.

[Cle86] Richard Cleve. "Limits on the Security of Coin Flips when Half the Processors Are Faulty (Extended Abstract)". In: *18th ACM STOC*. ACM Press, May 1986, pp. 364–369. DOI: 10.1145/12130.12168.

[Dam+21] Ivan Damgård et al. "Two-Round n-out-of-n and Multi-signatures and Trapdoor Commitment from Lattices". In: *PKC 2021, Part I*. Ed. by Juan Garay. Vol. 12710. LNCS. Springer, Cham, May 2021, pp. 99–130. DOI: 10.1007/978-3-030-75245-3_5.

[DEF18] Stefan Dziembowski, Lisa Eckey, and Sebastian Faust. "FairSwap: How To Fairly Exchange Digital Goods". In: *ACM CCS 2018*. Ed. by David Lie et al. ACM Press, Oct. 2018, pp. 967–984. DOI: 10.1145/3243734.3243857.

[DMP88] Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. "Non-Interactive Zero-Knowledge Proof Systems". In: *CRYPTO'87*. Ed. by Carl Pomerance. Vol. 293. LNCS. Springer, Berlin, Heidelberg, Aug. 1988, pp. 52–72. DOI: 10.1007/3-540-48184-2_5.

[DOY22] Wei Dai, Tatsuaki Okamoto, and Go Yamamoto. "Stronger Security and Generic Constructions for Adaptor Signatures". In: *INDOCRYPT 2022*. Ed. by Takanori Isobe and Santanu Sarkar. Vol. 13774. LNCS. Springer, Cham, Dec. 2022, pp. 52–77. DOI: 10.1007/978-3-031-22912-1_3.

[EFS20] Lisa Eckey, Sebastian Faust, and Benjamin Schlosser. "OptiSwap: Fast Optimistic Fair Exchange". In: *ASIACCS 20*. Ed. by Hung-Min Sun et al. ACM Press, Oct. 2020, pp. 543–557. DOI: 10.1145/3320269.3384749.

[Erw+21] Andreas Erwig et al. "Two-Party Adaptor Signatures from Identification Schemes". In: *PKC 2021, Part I*. Ed. by Juan Garay. Vol. 12710. LNCS. Springer, Cham, May 2021, pp. 451–480. DOI: 10.1007/978-3-030-75245-3_17.

[Fis05] Marc Fischlin. "Communication-Efficient Non-interactive Proofs of Knowledge with Online Extractors". In: *CRYPTO 2005*. Ed. by Victor Shoup. Vol. 3621. LNCS. Springer, Berlin, Heidelberg, Aug. 2005, pp. 152–168. DOI: 10.1007/11535218_10.

[FY92] Matthew K. Franklin and Moti Yung. "Communication Complexity of Secure Computation (Extended Abstract)". In: *24th ACM STOC*. ACM Press, May 1992, pp. 699–710. DOI: 10.1145/129712.129780.

[Gad+23] Sivanarayana Gaddam et al. *How to Design Fair Protocols in the Multi-Blockchain Setting*. Cryptology ePrint Archive, Report 2023/762. 2023. URL: https://eprint.iacr.org/2023/762.

[Gar+13] Sanjam Garg et al. "Witness encryption and its applications". In: *45th ACM STOC*. Ed. by Dan Boneh, Tim Roughgarden, and Joan Feigenbaum. ACM Press, June 2013, pp. 467–476. DOI: 10.1145/2488608.2488667.

[Ger+24a] Paul Gerhart et al. "Foundations of Adaptor Signatures". In: *Progress in Cryptology—Eurocrypt 2024*. Springer, 2024, pp. XXX–XXX.

[Ger+24b] Paul Gerhart et al. "Foundations of Adaptor Signatures". In: *EUROCRYPT 2024, Part II*. Ed. by Marc Joye and Gregor Leander. Vol. 14652. LNCS. Springer, Cham, May 2024, pp. 161–189. DOI: 10.1007/978-3-031-58723-8_6.

[GK10] S. Dov Gordon and Jonathan Katz. "Partial Fairness in Secure Two-Party Computation". In: *EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. LNCS. Springer, Berlin, Heidelberg, 2010, pp. 157–176. DOI: 10.1007/978-3-642-13190-5_8.

[Ji+24] Yunfeng Ji et al. "Threshold/Multi Adaptor Signature and Their Applications in Blockchains". In: *Electronics* 13.1 (2024). ISSN: 2079-9292. DOI: 10.3390/electronics13010076. URL: https://www.mdpi.com/2079-9292/13/1/76.

[Kat24] Jonathan Katz. "Round-Optimal, Fully Secure Distributed Key Generation". In: *CRYPTO 2024, Part VII*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14926. LNCS. Springer, Cham, Aug. 2024, pp. 285–316. DOI: 10.1007/978-3-031-68394-7_10.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. 2nd. Chapman & Hall/CRC, 2014. ISBN: 1466570261.

[KMB15] Ranjit Kumaresan, Tal Moran, and Iddo Bentov. "How to Use Bitcoin to Play Decentralized Poker". In: *ACM CCS 2015*. Ed. by Indrajit Ray, Ninghui Li, and Christopher Kruegel. ACM Press, Oct. 2015, pp. 195–206. DOI: 10.1145/2810103.2813712.

[KOT24] Kaisei Kajita, Go Ohtake, and Tsuyoshi Takagi. *Consecutive Adaptor Signature Scheme: From Two-Party to N-Party Settings*. Cryptology ePrint Archive, Paper 2024/241. https://eprint.iacr.org/2024/241. 2024. URL: https://eprint.iacr.org/2024/241.

[KW03] Jonathan Katz and Nan Wang. "Efficiency Improvements for Signature Schemes with Tight Security Reductions". In: *ACM CCS 2003*. Ed. by Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger. ACM Press, Oct. 2003, pp. 155–164. DOI: 10.1145/948109.948132.

[Lin17] Yehuda Lindell. "Fast Secure Two-Party ECDSA Signing". In: *CRYPTO 2017, Part II*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10402. LNCS. Springer, Cham, Aug. 2017, pp. 613–644. DOI: 10.1007/978-3-319-63715-0_21.

[Poe17] Andrew Poelstra. "Scriptless scripts". In: *Presentation Slides* (2017).

[Sah99] Amit Sahai. "Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security". In: *40th FOCS*. IEEE Computer Society Press, Oct. 1999, pp. 543–553. DOI: 10.1109/SFFCS.1999.814628.

[Sch91] Claus-Peter Schnorr. "Efficient Signature Generation by Smart Cards". In: *Journal of Cryptology* 4.3 (Jan. 1991), pp. 161–174. DOI: 10.1007/BF00196725.

[Suw+22] Misni Harjo Suwito et al. "A Systematic Study of Bulletin Board and Its Application". In: *ASIACCS 22*. Ed. by Yuji Suga et al. ACM Press, 2022, pp. 1213–1215. DOI: 10.1145/3488932.3527280.

[Wat05] Brent R. Waters. "Efficient Identity-Based Encryption Without Random Oracles". In: *EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. LNCS. Springer, Berlin, Heidelberg, May 2005, pp. 114–127. DOI: 10.1007/11426639_7.

A Preliminaries

A.1 Hard Relations

A relation, denoted $\mathcal{R}$, is a function that maps $\mathcal{R} : \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}} \to \{0, 1\}$, where $\mathcal{D}_{\mathsf{S}}$ represents the set of all possible statements and $\mathcal{D}_{\mathsf{W}}$ the set of all potential witnesses. Consider a statement Y belonging to the space $\mathcal{D}_{\mathsf{S}}$, and a witness y from the space $\mathcal{D}_{\mathsf{W}}$. The relation $\mathcal{R}$ assigns the pair (s, w) the value 1 if and only if w serves as a valid witness for the statement s.

A relation is said to be hard if, given only the statement s, it is computationally hard to find a witness w that satisfies the relation (i.e., makes $\mathcal{R}(s, w) = 1$). This feature is crucial for certain applications.

Two conditions are essential for practicality: first, verifying that a given witness statement pair is valid (i.e., confirming that $\mathcal{R}(s, w) = 1$ for a given s and w) should be computationally easy. Second, generating instances of the relation - pairs (s, w) where w is a witness for s - should also

be computationally easy. These conditions ensure that while it is hard to find a witness for a given proposition, it is not hard to check the validity of a given witness-proposition pair and to generate examples of valid pairs.

Definition 17 (Hard Relation [Ger+24b])**.** *Let $\mathcal{R} \subseteq \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ with statement/witness pairs $(s, w) \in \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ be a relation and let the language $\mathcal{L}_{\mathcal{R}} \subseteq \mathcal{D}_{\mathsf{S}}$ associated to this relation $\mathcal{R}$ be defined as $L_{\mathcal{R}} := \{s \in \mathcal{D}_{\mathsf{S}} \mid \exists\, w \in \mathcal{D}_{\mathsf{W}} \text{ s.t. } (s, w) \in \mathcal{R}\}$. We say that $\mathcal{R}$ is a hard relation if:*

Sampling $(s, w) \leftarrow \mathsf{genR}(\lambda)$. There exists a PPT sampling *algorithm $\mathsf{genR}(\lambda)$ that on input the security parameter λ outputs a pair $(s, w) \in \mathcal{R}$.*

Decidability $\mathcal{R}(s, w)$. The relation $\mathcal{R}(s, w)$ is poly-time decidable*.*

Hardness. An efficiently sampable and decidable relation $\mathcal{R}$ is hard*, if for all* PPT *adversaries $\mathcal{A}$, the following probability is negligible:*

$$\Pr[\mathcal{R}(s, w^*) = 1 \mid (s, w) \leftarrow \mathsf{genR}(\lambda); w^* \leftarrow \mathcal{A}(s)] \leq \mathsf{negl}(\lambda)(1^\lambda),$$

where the probability is taken over the random choice of genR *and $\mathcal{A}$.*

To construct CWE encryption schemes, we require hard relations where sampling is not performed uniformly at random but instead depends on auxiliary information. We model this using a sampling algorithm genR' that takes as input the security parameter λ and an auxiliary input x. In the absence of auxiliary information, genR' reduces to the standard sampling algorithm for hard relations. We impose the usual requirements of decidability and hardness, ensuring that these properties hold even when an adversary provides auxiliary information. More formally:

Definition 18 (Hard Relation with Auxiliary Information)**.** *Let $\mathcal{R} \subseteq \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ with statement/witness pairs $(s, w) \in \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ be a relation and let the language $\mathcal{L}_{\mathcal{R}} \subseteq \mathcal{D}_{\mathsf{S}}$ associated to this relation $\mathcal{R}$ be defined as $L_{\mathcal{R}} := \{s \in \mathcal{D}_{\mathsf{S}} \mid \exists\, w \in \mathcal{D}_{\mathsf{W}} \text{ s.t. } (s, w) \in \mathcal{R}\}$. We say that $\mathcal{R}$ is a hard relation with auxiliary input if:*

Sampling $(s, w) \leftarrow \mathsf{genR}'(\lambda, x)$. There exists a PPT *sampling algorithm $\mathsf{genR}(\lambda)$ that on input the security parameter λ and auxiliary input $x \in \{0,1\}^\lambda$ outputs a pair $(s, w) \in \mathcal{R}$.*

Decidability $\mathcal{R}(s, w)$. The relation $\mathcal{R}(s, w)$ is poly-time decidable*.*

Hardness. An efficiently sampable and decidable relation $\mathcal{R}$ is hard*, if for all* PPT *adversaries $\mathcal{A}$, the following probability is negligible:*

$$\Pr[\mathcal{R}(s, w^*) = 1 \mid x^* \leftarrow \mathcal{A}(\lambda); (s, w) \leftarrow \mathsf{genR}(\lambda, x^*); w^* \leftarrow \mathcal{A}(s, x^*)] \leq \mathsf{negl}(\lambda),$$

where the probability is taken over the random choice of genR *and $\mathcal{A}$.*

A.2 Adaptor Signatures

We begin by recalling the definition of (standard) adaptor signatures [Erw+21; Aum+21]. An adaptor signature scheme, defined for a hard relation $\mathcal{R}$ and signature scheme Σ, consists of four algorithms. The pre-signing algorithm links a statement s from $\mathcal{R}$ and a message msg to a publicly verifiable pre-signature $\widetilde{\sigma}$. It ensures that anyone with the correct witness w can convert $\widetilde{\sigma}$ to a full signature σ using the Adapt method. In addition, possession of σ and $\widetilde{\sigma}$ allows efficient extraction of the witness w using the Extract algorithm.

Definition 19 (Adaptor Signature)**.** *An* adaptor signature scheme *w.r.t. a hard relation* $\mathcal{R}$ *and a signature scheme* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *consists of a tuple of four algorithms* $\mathsf{AS}_{\mathcal{R},\Sigma} = (\mathsf{pSign}, \mathsf{Adapt}, \mathsf{pVrfy}, \mathsf{Extract})$ *defined as:*

$\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, s)$*. The* pre-signing *algorithm is a* PPT *algorithm that on input a secret key* sk*, message* $m \in \{0,1\}^{l_m}$ *and statement* $s \in L_{\mathcal{R}}$*, outputs a pre-signature* $\widetilde{\sigma}$*.*

$b \leftarrow \mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma})$*. The* pre-verification *algorithm is a* DPT *algorithm that on input a public key* pk*, message* $m \in \{0,1\}^{l_m}$*, statement* $s \in L_{\mathcal{R}}$ *and pre-signature* $\widetilde{\sigma}$*, outputs a bit* b*.*

$\sigma \leftarrow \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w)$*. The* adapting *algorithm is a* PPT *algorithm that on input a pre-signature* $\widetilde{\sigma}$ *and witness* w *for the statement* $s \in L_{\mathcal{R}}$ *outputs an adapted signature* σ*.*

$w \leftarrow \mathsf{Extract}(\mathsf{pk}, \widetilde{\sigma}, \sigma, s)$*. The* extracting *algorithm is a* DPT *algorithm that on input a pre-signature* $\widetilde{\sigma}$*, signature* σ *and statement* $s \in L_{\mathcal{R}}$*, outputs a witness* w *such that* $(s, w) \in \mathcal{R}$*, or* $\perp$*.*

Definition 20 (Pre-signature correctness)**.** *An adaptor signature* $\mathsf{AS}_{R,\Sigma}$ *satisfies* pre-signature correctness*, if for all* $\lambda \in \mathbb{N}$ *and* $m \in \{0,1\}^{l_m}$*:*

$$\Pr\left[\begin{array}{c|l} \mathsf{pVrfy}(\mathsf{pk}, m, s, \widetilde{\sigma}) = 1 \wedge & (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda), \\ \mathsf{Vrfy}(\mathsf{pk}, m, \sigma) = 1 \wedge & (s, w) \leftarrow \mathsf{genR}(1^\lambda), \\ (s, w') \in \mathcal{R} & \widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, s) \\ & \sigma := \mathsf{Adapt}(\mathsf{pk}, \widetilde{\sigma}, w), \\ & w' := \mathsf{Extract}(\mathsf{pk}, \widetilde{\sigma}, \sigma, s) \end{array}\right] = 1.$$

For the security properties of adaptor signatures, we refer to [Ger+24b].

A.3 Threshold Signatures

Threshold signatures are a type of 't-out-of-n' signature: They require a key generation process that involves n participants, and after this, any subset of t members (where n and t are predetermined parameters set during the initial key generation) can collaboratively generate a valid signature under a collective public key. This public key represents the entire group of n participants. Furthermore, unforgeability guarantees that even a group of up to $t-1$ malicious signers cannot produce a forged signature.

Definition 21 (Threshold Signature Scheme [Dam+21; Lin17])**.** *A* threshold signature scheme $\mathsf{TS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *consists of algorithms and protocols as follows:*

$par \leftarrow \mathsf{Setup}(n, t)$*: The* setup algorithm Setup *takes as input the number* n *of signers and the signing threshold* t*, and outputs public parameters* par*. From now on,* par *is an implicit input to all subsequent algorithms.*

$(\mathsf{pk}, \{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}) \leftarrow \mathsf{KeyGen}(par)$*: The* key generation algorithm KeyGen *takes as input the public parameters* par *and outputs a combined public key* pk*, and a signing key share* sk_i *for each signer* $\mathcal{S}_i$*.*

$\sigma \leftarrow \langle \mathsf{Sign}(\mathsf{sk}_i, m) \rangle$*: The* signing protocol Sign *is an interactive algorithm of which an instance is run by each signer* $\mathcal{S}_1, \ldots, \mathcal{S}_n$ *concurrently. Concretely, signer,* $\mathcal{S}_i$ *runs* Sign_i*, which takes as input a secret key share* sk_i*, and a message* m*. At the end of the protocol,* $\mathcal{S}_i$ *obtains a signature* σ *as output.*

$\underline{b \leftarrow \mathsf{Verify}(\mathsf{pk}, m, \sigma)}$*: The verification algorithm takes as input a public key* pk*, a message* m*, and a signature* σ*. It outputs a boolean* b*, where* $b = \mathbf{true}$ *means that the signature is valid and* **false** *that it is invalid.*

CORRECTNESS. A threshold signature scheme is correct if for all $\lambda, t \leq nin\mathbb{N}$, all messages $m \in \{0,1\}^*$, for all keys $(\mathsf{pk}, \{\mathsf{sk}_1, \ldots, \mathsf{sk}_n\}) \leftarrow \mathsf{KeyGen}(par)$, each $\mathsf{hon} \subset \{1, \ldots, n\}$ with $|\mathsf{hon}| \geq t$, each output of the signing protocol also verifies, i.e.,

$$\mathsf{Vrfy}(\mathsf{pk}, m, \langle \{\mathsf{Sign}(\mathsf{sk}_i, m)\}_{i \in \mathsf{hon}} \rangle) = 1.$$

Construction 3 (Sparkle Schnorr Threshold Signature [CKM23])**.** *We construct the sparkle Schnorr threshold signature scheme using the algorithms* $\mathsf{Setup}, \mathsf{KGen}, \mathsf{Sign}^0, \mathsf{Sign}^1, \mathsf{Sign}^2, \mathsf{Combine}, \mathsf{Vrfy}$*, which we provide in Figure 12.*

A.4 Non-Interactive Zero-Knowledge Proofs

Let $\mathcal{R} : b \leftarrow \{0,1\}^* \times \{0,1\}^*$ be a NP-witness-relation with NP-language $\mathcal{L}_{\mathcal{R}} := \{x \mid \exists w : \mathcal{R}(x,w) = 1\}$. A non-interactive zero-knowledge proof (NIZK) system [DMP88; Fis05] for the relation R consists of the three efficient algorithms $crs \leftarrow\!\$\ \mathsf{Setup}(1^\lambda), \pi \leftarrow\!\$\ \mathsf{P}(crs, x, w)$, and $b \leftarrow \mathsf{V}(crs, x, \pi)$. The setup algorithm inputs the security parameter and returns a common reference string. The prove algorithm takes as input crs and a statement x with a witness w and generates a proof π. The verify algorithm takes as input crs, x, and π and returns $b = 1$ if the proof is valid and $b = 0$ otherwise. We use a NIZK system that is:

- complete, requiring that for every $x \in L$, P generates proofs that V accepts $Pr[\mathsf{V}(crs, x, \pi) = 1 | crs \leftarrow\!\$\ \mathsf{Setup}(1^\lambda), \pi \leftarrow\!\$\ \mathsf{P}(crs, x, w)] \geq 1 - \mathsf{negl}(\lambda) \forall x \in L$,
- sound, meaning that for every $x \notin L$, it is hard for an adversary to generate proofs that V accepts $Pr[\mathsf{V}(crs, x, \pi) = 1 | crs \leftarrow\!\$\ \mathsf{Setup}(1^\lambda), \pi \leftarrow\!\$\ \mathsf{P}(crs, x, w)] \leq \mathsf{negl}(\lambda) \forall x \notin L$,,
- simulation sound [Sah99], meaning that the system is sound even after the adversary has seen simulated proofs of its choice, and
- zero-knowledge, ensuring that the verifier learns nothing beyond the validity of the statement x This property is proved by constructing a simulator Sim that computes proofs without knowing the witness (but with some trapdoors) that are indistinguishable from "real" proofs.

$$\left| \Pr\left[\mathcal{A}^{\mathsf{Sim}(crs,x)} = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{V}(crs,x,w)} = 1\right] \right| \leq \mathsf{negl}(\lambda).$$

For formal security definitions, we refer the reader to [DMP88; Fis05].

A.5 (Verifiable) Public Key Encryption Schemes

We recite the definition of a public key encryption scheme.

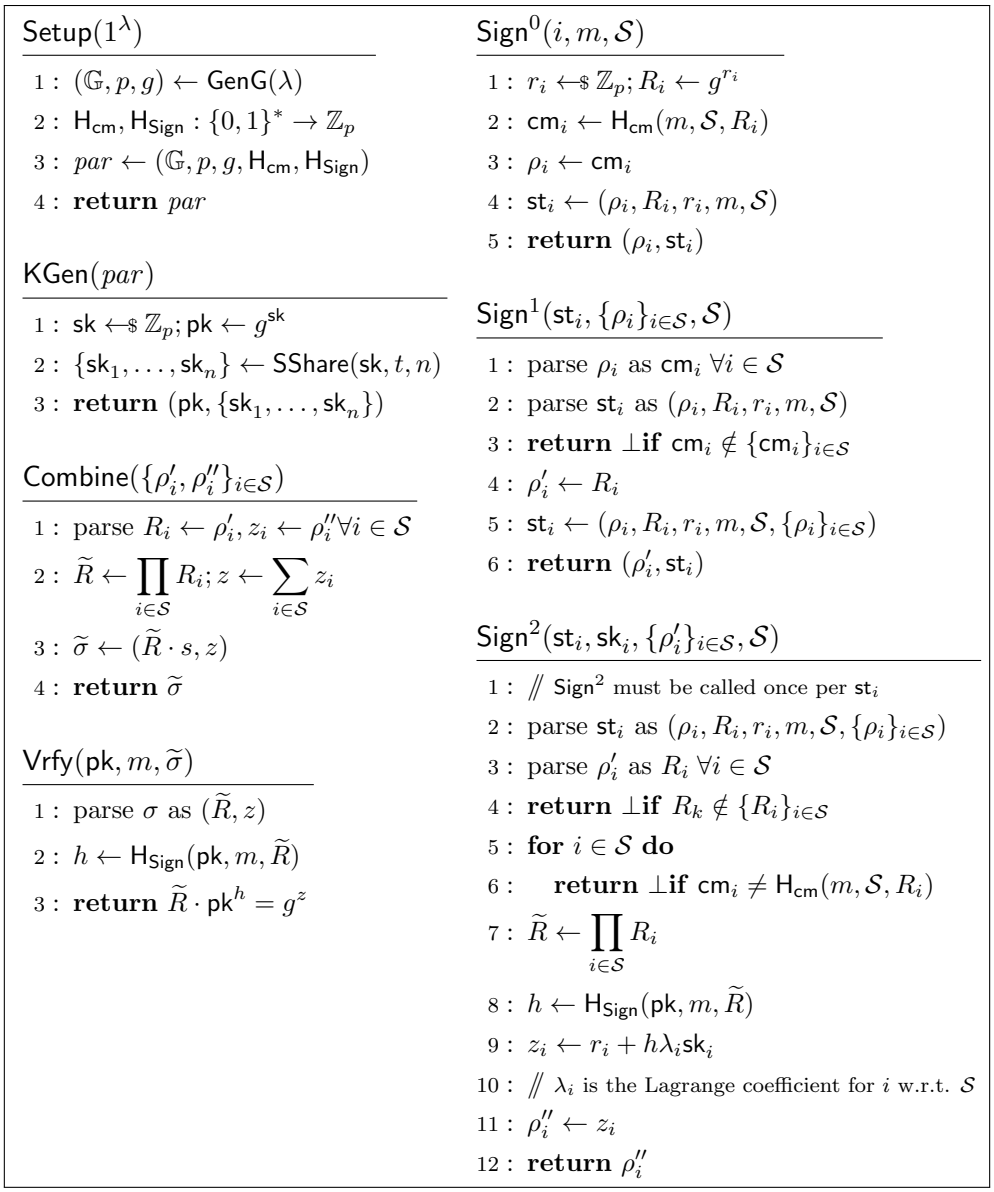

Figure 12: Algorithms of Sparkle [CKM23].

Definition 22 (Public-Key Encryption Scheme [KL14])**.** *A* public-key encryption scheme $\Pi = (\mathsf{KGen}, \mathsf{Enc}, dec)$ *consists of three* PPT *algorithms*

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda)$*: The* key generation *algorithm outputs a pair* (sk, pk)*. We refer to the first of these as the* private key *and the second as the* public key*.*

$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$*: The* encryption *algorithm takes as input a public-key* pk *and a plaintext message* m *and outputs a ciphertext* c*.*

$m =: \mathsf{Dec}(\mathsf{sk}, c)$*: The* decryption *algorithm takes as input a private-key* sk *and a ciphertext* c *and outputs some plaintext* m*.*

We say a public key encryption scheme Π verifiably [CD00] encrypts a message $m \in \mathbb{Z}_p$ w.r.t. a group $\mathbb{G}$ of order p (p is a prime), if the final ciphertext consists of the ciphertext c of the encryption scheme, a statement $M = g^m$ and a NIZK, which proves that c arises by encrypting the value m, which corresponds to the statement M. If we want to use such a generic verifiable encryption, we use the canonical algorithms $\mathsf{VPKE.Enc}, \mathsf{VPKE.Vrfy}$ to denote encryption and canonical ciphertext verification.

B Additional Proofs

In this section, we provide deferred proofs for threshold adaptor signatures, multi-party fair exchange protocols, and certified witness encryption.

B.1 Proofs for Threshold Adaptor Signatures

Lemma 3 (Unique Extractability)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. Then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves extractability in the random oracle model.*

Proof of Lemma 3. To prove the unique extractability of Construction 1, we show that it is impossible to construct two distinct signatures, which both extract with the same pre-signature into a valid witness. The first observation of this impossibility is that sparkle is defined w.r.t. a bijective hard relation. I.e., each statement s corresponds to exactly one single witness s. Therefore, if there exist two distinct signatures $\sigma = (\sigma_1, \sigma_2) \neq \sigma' = (\sigma'_1, \sigma'_2)$ that both extract with a pre-signature $\widetilde{\sigma} = (\widetilde{\sigma_1}, \widetilde{\sigma_2})$, then it holds, that :

$$\sigma_2 - \widetilde{\sigma_2} = w = \sigma'_2 - \widetilde{\sigma_2},$$

which implies that $\sigma_2 = \sigma'_2$. But if $\sigma_2 = \sigma'_2$, then also $\sigma_1 = \sigma'_1$, as the relation is bijective, and the random oracle model is collision-free. This implies that the winning condition of unique extractability cannot be met for Construction 1. □

Lemma 4 (Unlinkability)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. Then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves unlinkability.*

Proof. To show the unlinkability of Construction 1, we show that adapted pre-signatures, even in the presence of $t-1$ malicious signers, are identically distributed over ordinary signatures. Hence, no adversary can distinguish them. An adapted pre-signature for the statement-witness pair (s, w) has the form $(R \cdot s, r + y + \mathsf{sk} \cdot h)$ and an ordinary signature has the form $(R', r' + \mathsf{sk} \cdot h')$. Both Construction 1 and Sparkle have an initial commit round pSign^0, which guarantees that the final randomness r is used so the sign is uniformly distributed. If r is a uniformly random element, then so is $r + w$ for all w. Therefore, the distributions are equal, which implies that no adversary can distinguish adapted pre-signatures from ordinary signatures. □

Lemma 5 (Pre-Signature Adaptability)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. Then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves pre-signature adaptability.*

Proof. A Schnorr signature $\sigma = (R, z) = (g^r, \mathsf{sk} \cdot h + r)$ with $h = \mathsf{H}(\mathsf{pk}, R, m)$ verifies, if $\mathsf{pk}^h \cdot R = g^z$. A pre-signature $\widetilde{\sigma} = (R \cdot s, z) = (g^r, \mathsf{sk} \cdot h + r)$ with $h = \mathsf{H}(\mathsf{pk}, R \cdot s, m)$ of Construction 1 pre-verifies, if $\mathsf{pk}^h \cdot R \cdot s = g^z \cdot Y$. Therefore, each adapted pre-signature (i.e., $(R \cdot s, z + w)$) verifies if and only if it pre-verifies as pre-signature since $g^z \cdot s = g^{z+w}$. □

Lemma 6 (Computational Pre-Verify Soundness)**.** *Let* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *be defined as in Construction 1. Then* $\mathsf{TAS}_{\mathsf{Sparkle},\mathcal{R}}$ *achieves pre-verify soundness.*

Proof. Since the DLog relation is bijective; each element in the domain of the statements is also in the language of the relation. If a statement s is not in this domain, the multiplications $g^z \cdot s$ and $g^z \cdot w$ fail. Therefore, Construction 1 achieves computational pre-verify soundness. □

B.2 Proofs for MPFE

Lemma 7 (Seller Fairness)**.** *Let* $\mathcal{R}$ *be a hard relation for the DLog language in the group* $\mathbb{G}$*, and let* TAS *be the secure Schnorr threshold adaptor signature scheme defined in Construction 1. Then,Construction 1 achieves* t_S*-seller fairness if at least* t_S *honest sellers participate in the protocol in the presence of a minimal blockchain with timing and in the random oracle model.*

Proof of Lemma 7. To prove the seller fairness of Construction 1, we assume, towards contradiction, that there exists an efficient adversary $\mathcal{A}$ breaking the seller fairness. This means the adversary has to produce at least a single challenge witness, for which at least one honest seller is not compensated, or at least a single honest seller has fewer funds after the protocol execution than before. In our protocol, the buyer locks the input transaction for the fair exchange for a sufficient time. This means, that when a single honest seller receives the pre-signature and the blinded witness in time, such a seller can claim the payment. Since we assume t_S honest sellers participate in the selling protocol, and all honest sellers share their blinded witness share upon receiving the pre-signature, there will be at least a single honest seller who can combine the full blinded witness *before* the time-lock expires. This seller can then adapt the pre-signature. By the pre-signature adaptability and pre-verify soundness of the threshold adaptor signature scheme, this adapted pre-signature is valid w.r.t. pk_B. Since the coins are locked, the honest seller can post the adapted pre-signature on-chain. This means, the sellers can successfully claim the payment. Hence, we are always in a situation where all honest sellers are paid. Since no seller authorizes a payment, the second winning condition cannot be triggered. In addition, since the witness is information-theoretically hidden from the adversary, the adversary cannot learn the witness when no coin has been locked before. □

Lemma 8 (Witness Privacy)**.** *Let* $\mathcal{R}$ *be a hard relation for the DLog language in the group* $\mathbb{G}$ *whose witness is shared using Shamir secret sharing. Let* Π_{Enc} *be a CPA secure public key encryption scheme. Then, Construction 1 achieves witness privacy for any* $t_\mathsf{S} \leq n_\mathsf{S}$ *in the random oracle model.*

Proof of Lemma 8. To prove the witness privacy of Construction 1, we assume, towards a contradiction, that there exists an efficient adversary $\mathcal{A}$ that is capable of winning the WitPriv game with non-negligible probability. To win this game, $\mathcal{A}$ has to output a challenge witness, from which it knows $t_\mathsf{S} - 1$ shares. Since the Shamir secret sharing is information-theoretically secure, $\mathcal{A}$ can only

break the $\mathsf{WitPriv}$ game by learning the witness share of at least one honest seller. The honest sellers follow the construction Construction 1, and hence, only forward blinded witness-shares $w_i + e_i$ to $\mathcal{A}$. Since e_i is random (it is output by a random oracle and $\mathcal{A}$ does not know the random key k), the witness share is information-theoretically hidden from $\mathcal{A}$, as long as it does not know e_i. The adversary has three attack vectors to learn e_i. First, the adversary knows the statement $E_i = g^{e_i}$. Second, the adversary knows the CPA-secure encryption of e_i w.r.t. the honest seller's encryption key. Third, the adversary knows $t_{\mathsf{S}} - 1$ computations of e_i. We now build a series of game hops that show that the adversary's capabilities of winning the $\mathsf{WitPriv}$ game are negligible.

Game $\mathcal{G}_0$: The initial game $\mathcal{G}_0$ equals the $\mathsf{WitPriv}$ game and it holds that $\Pr[\mathsf{WitPriv}(\lambda) = 1] = \Pr[\mathcal{G}_0(\lambda) = 1]$.

Game $\mathcal{G}_1$: In the first game, we replace each honest signer's partial blinding factor e_j, which is encrypted using the CPA secure encryption scheme by a random value in each public state that is forwarded to the adversary.

Claim 3. *The games $\mathcal{G}_0$ and game $\mathcal{G}_1$ have a negligible transition of* $\Pr[\mathcal{G}_0(\lambda) = 1] \leq \Pr[\mathcal{G}_1(\lambda) = 1] + \mathsf{negl}_1(\lambda)$ *due to the CPA security of the encryption scheme.*

Proof. The proof of this claim equals the CPA security game-hop proof of Lemma 7. □

In the situation of game $\mathcal{G}_1$, the adversary has no knowledge of the random element e_i. Therefore, the honest witness share w_i is information-theoretically hidden from $\mathcal{A}$. The winning probability of $\mathcal{A}$ in winning the game $\mathcal{G}_1$ is exactly as in breaking the hardness of the relation $\mathcal{R}$. The winning probability of $\mathcal{A}$ when brute-forcing the key k is also negligible since the value e_i is computed by a random oracle and the key k is chosen uniformly from an exponential space. Therefore, no efficient adversary against the witness privacy of Construction 1 exists. □

B.3 Proofs for Certified Witness Encryption

Lemma 9. *The Schnorr* CWE *scheme in Construction 2 is correct.*

Proof. To verify correctness of Construction 2, we have to check, that for all messages m and statement-witness pairs $((\mathsf{vk}, m, R), (R, z)) \in \mathcal{R}$, with $\mathsf{vk} = g^{\mathsf{sk}}$ and $R = g^r$ it holds that $m = \mathsf{Dec}_{\mathsf{CWE}}(w, \mathsf{Enc}_{\mathsf{CWE}}(\mathsf{ek}, m))$. If $\sigma = (R, z)$ is a valid signature w.r.t. (vk, m) it holds that $\mathsf{vk}^h \cdot R = g^z$ for $h = \mathsf{H}(\mathsf{vk}||R||m)$. Therefore, we have that for all (c_1, c_2) output by $\mathsf{Enc}_{\mathsf{CWE}}$, that

$$\frac{c_2}{c_1^z} = \frac{\mathsf{ek}^y \cdot m}{g^{y \cdot z}} = \frac{g^{(\mathsf{sk} \cdot h + r) \cdot y} \cdot m}{g^{y \cdot (\mathsf{sk} \cdot h + r)}} = m.$$

Henceforth, Construction 2 is correct. □

Proof of Theorem 2. By contradiction, we assume that there exists an adversary $\mathcal{A}$ that breaks the CPA security of Π_{CWE} with non-negligible probability. We use this adversary to build a reduction from the CPA security of Π_{CWE} to the hardness of DDH. The reduction $\mathcal{R}$ gets as input the

values $([\alpha], [\beta], [\gamma])$, such that γ is either a random value or $\alpha \cdot \beta$. The reduction forwards $[\alpha]$ as verification key to $\mathcal{A}$, which eventually outputs three messages (m_i, m_1, m). $\mathcal{R}$ samples a fresh randomness, commitment pair $(R, r) \leftarrow \mathsf{genR}(1^\lambda)$ and computes the challenge ciphertext on the challenge message m_b via

$$c_b \leftarrow ([\beta], [\beta]^r \cdot [\gamma]^h \cdot m_b).$$

The reduction forwards c_b to $\mathcal{A}$, which eventually outputs a bit b' and $\mathcal{R}$ returns $b' == b$.

Claim 4. *If $\gamma \neq \alpha \cdot \beta$, then $\mathcal{A}$ wins with probability at most $1/2$.*

Proof. If $\gamma \neq \alpha \cdot \beta$, then γ is a random element in $\mathbb{Z}_p$. Thus, the challenge ciphertext c_b is also a random element in $\mathbb{Z}_p$ (as H is a non-zero function) and is independent of the challenge message m_b. So, the claim follows. □

Claim 5. *If $\gamma = \alpha \cdot \beta$, then $\mathcal{A}$ wins with non-negligible advantage ε.*

Proof. If $\gamma = \alpha \cdot \beta$, then the ciphertext is exactly constructed as the adversary expects it to be:

$$\begin{aligned} c := ([\beta], [\beta]^r \cdot [\gamma]^h \cdot m_b) &= ([\beta], [\beta \cdot r + \alpha \cdot \beta \cdot h] \cdot m_b) \\ &= ([\beta], [\beta \cdot (r + \alpha \cdot h)] \cdot m_b). \end{aligned}$$

As we assume by contradiction that $\mathcal{A}$ breaks the CPA security of the encryption scheme with non-negligible probability, this probability carries over if $\gamma = \alpha \cdot \beta$. □

With the standard argument for the CPA security of the original Elgamal encryption, it holds that

$$\begin{aligned} \left|\Pr\left[\mathsf{DDH}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{DDH}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \\ \left|\Pr\left[\mathsf{CPA}^1_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{CPA}^0_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \\ \left|\frac{1}{2} + \varepsilon - \frac{1}{2}\right| \geq \varepsilon. \end{aligned}$$

This contradicts the hardness of DDH; thus, no efficient adversary against the CPA security of Construction 2 can exist.

□

C Threshold Adaptor Sigantures

We provide the missing algorithms for threshold adaptor signatures in Figure 13.

D A Framework for CWE

A detailed examination of the DDH-based construction described in Sec. 5 identifies certain properties that are essential for developing a highly efficient CWE. In this section, we will abstract

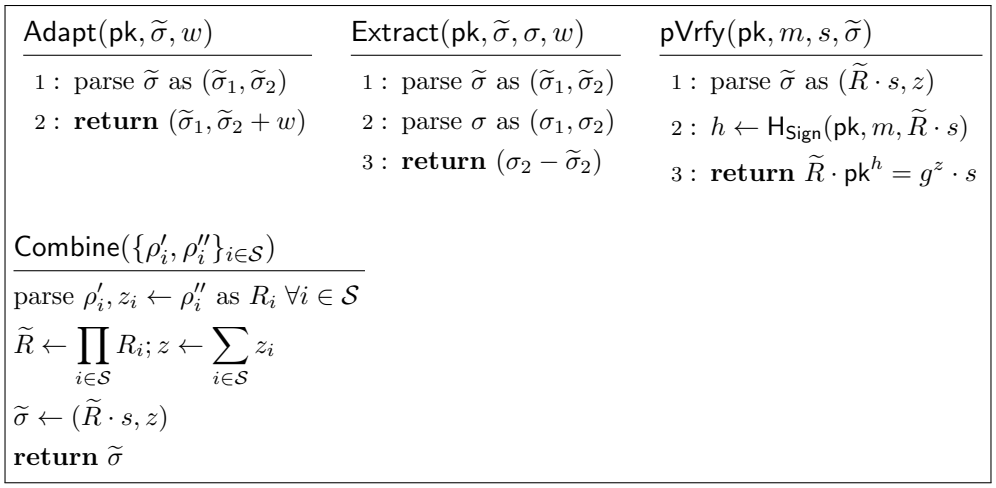

Figure 13: The algorithms Adapt, Extract, pVrfy, and Combine

these properties with the goal of constructing CPA-secure CWE for a large class of signatures. We refer to this class of signatures as "certificate-extractable" signatures and provide a construction that leverages the properties of certificate-extractable signatures into a CWE scheme. Our generalization is general enough to encompass constructions that are secure under the CDH assumption, which is of particular interest in the context of bilinear-pairing friendly groups. As for concrete instances, we show that our framework is compatible with the Schnorr [Sch91], the BBS+ [BBS04] the Camenisch-Lysyanskaya (CL) [CL03], (Waters+) [BSW06], and Katz-Wang [KW03] signature schemes.

In light of our warmup example, three principal properties inherent to the signature scheme are necessary for the construction of a secure CWE scheme. First, it is necessary to establish a methodology for the computation of an encryption key derived from the tuple (vk, m, and R) and a decryption key derived from the corresponding signature (σ). This property is referred to as *certifiability*. Secondly, it is necessary that there are not exponentially many signatures that can serve as a decryption key; instead, only the signature generated with the correct randomness can be used. Therefore, it is essential that we can extract the commitment, denoted as *commitment extractability*, from a signature. More formally, we define certificate-extractable signatures as follows.

Definition 23 (Certificate-Extractable Signature)**.** *Let* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a randomized signature scheme with randomness space* $\mathbb{Z}_p$*. We denote* R *as the commitment of a signature's randomness,* vk *as a verification key from the signature schemes keyspace, and* m *a message from the message space. We say* Σ *is* certificate extractable *if the following conditions hold.*

Certifiability: There exist deterministic functions PrivC *and* PubC*. The function* PrivC *extracts a certificate from a signature (i.e.,* $c \leftarrow \mathsf{PrivC}(\sigma)$*). The function* PubC *computes a commitment to a signatures certificate using public inputs (i.e.,* $C \leftarrow \mathsf{PubC}(\mathsf{vk}, m, R)$*).*

One-Wayness: It holds, that $\mathsf{PubC}(\mathsf{vk}, m, R) = [\mathsf{PrivC}(\sigma)]$*.*

Commitment Extractability: There exists an efficient algorithm ExtR *that on input a signature* σ *and a verification key* vk *outputs a commitment* R *to the signature's randomness.*

CWE FOR CERTIFICATE-EXTRACTABLE SIGNATURES. The use of certificate-extractable signatures necessitates the definition of a language for an underlying hard relation that ensures that not every valid signature on a message can be decrypted, but only those signatures that are based on the correct randomness. To construct such a language, we generalize the language from the Schnorr setting and utilize the commitment extractability of the certificate-extractable signature scheme. More formally, we define a statement of $\mathcal{L}_{\mathcal{R}}$ as a tuple of the verification key vk, message m, and commitment to a randomness R and a witness a valid signature w.r.t. (vk, m) which extracts the commitment R leading to the language

$$\mathcal{L}_{\mathcal{R}} := \{((\mathsf{vk}, m, R), \sigma) \text{ s.t. } \mathsf{Vrfy}(\mathsf{vk}, \sigma, m) = 1 \wedge \mathsf{ExtR}(\sigma) = R\}.$$

Using this language, we now define the CWE scheme for certificate-extractable signatures.

Construction 4 (CWE for CE Signatures)**.** *Let $\mathbb{G}$ be a group of prime order p. Let H be a function mapping into $\mathbb{G}$. We define the* CWE *scheme $\Pi_{\mathsf{CWE}} = (\mathsf{KG}_{\mathsf{CWE}}, \mathsf{Enc}_{\mathsf{CWE}}, \mathsf{Dec}_{\mathsf{CWE}})$ for certificate extractable signatures w.r.t. $(\mathbb{G}, p)$ in Figure 14.*

$\mathsf{KG}_{\mathsf{CWE}}(\mathsf{vk}, m, R)$	$\mathsf{Dec}_{\mathsf{CWE}}(\sigma, c)$	$\mathsf{Enc}_{\mathsf{CWE}}(\mathsf{ek}, m)$
1 : $\mathsf{ek} \leftarrow \mathsf{PubC}(\mathsf{vk}, m, R)$	1 : parse c as (c_1, c_2)	1 : $a \leftarrow\!\$\ \mathbb{Z}_p$
2 : **return** ek	2 : $\mathsf{dk} \leftarrow \mathsf{PrivC}(\sigma)$	2 : $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{ek}^a) \cdot m$
	3 : **return** $c_2 / \left(\mathsf{H}(c_1^{\mathsf{dk}})\right)$	3 : **return** $([a], h)$

Figure 14: CWE encryption for certificate-extractable signatures.

CWE FROM DH. In this section, we show the CPA security of Construction 4 based on the decisional Diffie-Hellman (DDH) and the computational Diffie-Hellman (CDH) assumptions.

Looking ahead to the proof of the CPA security of Construction 4 w.r.t. the DDH/CDH assumptions, we first formalize a property that the signature scheme needs to achieve for our proof to simulate the challenge ciphertexts properly. We refer to this property as DH simulatability. On a high level, DH simulatability guarantees, that given a DH tuple $[\alpha], [\beta]$, our reduction can come up with a verification key vk, a random commitment R, such that for any message m, our reduction can simulate the value $[\delta] = C^{\alpha}$ for $C \leftarrow \mathsf{PubC}(\mathsf{vk}, m, R)$ using the factors $[\alpha], [\beta]$. Looking ahead, this will allow us to reduce the security of our reduction to both the CDH and the DDH assumption.

Definition 24 (DH Simulatability)**.** *We say a certificate-extractable signature scheme Σ is* DH-simulatable*, if for any $\beta \leftarrow\!\$\ \mathbb{Z}_p$, there exists an algorithm* SimDDH*, that on input $[\beta]$ outputs a set $(\mathsf{vk}, R, x_0, x_1)$, such that vk is in the domain of Σ's verification keys, $R = \mathsf{OWF}(r)$ is a commitment to a signatures randomness r, $x_0, x_1 \in \mathbb{Z}_p$, and $x_0 \neq 0$. In addition, for all m in the message space, the value $C \leftarrow \mathsf{PubC}(\mathsf{vk}, m, R)$ can be expressed via $C = x_0[\beta] + x_1$.*

If $C = [c]$ can be represented via $x_0[\beta] + x_1$, we can build the value $[\alpha \cdot c]$ via $[\alpha \cdot c] = [\alpha(x_0\beta + x1)] = x_0[\alpha\beta] + x_1[\alpha]$. Thus, if we are able to find $[\alpha \cdot c]$, we can compute $[\alpha\beta]$ via $[\alpha \cdot \beta] = \frac{[\alpha \cdot c] - x_1[\alpha]}{x_0}$ solving the CDH instance. In addition, given a DDH triple $([\alpha], [\beta], [\gamma])$, we can express $[\alpha \cdot c]$ by non-trivially using $[\gamma]$, allowing us to reduce the CPA security of CWE to DDH. Using DH simulatability, we now state the CPA security of Construction 4 based on both

DDH and CDH for DH-simulatable certificate-extractable signature schemes. We defer the proof to Appendix B.3. In addition, we defer the proof of CWE from DDH, and instantiations of our framework to Appendix D.

Theorem 6 (CPA Secure CWE from DDH)**.** *Let Σ be a certificate-extractable signature scheme with DH-simulatability w.r.t. a one-way function OWF. If the decisional Diffie-Hellman (DDH) assumption is hard in $\mathbb{G}$ w.r.t. OWF, and H is the identity in $\mathbb{G}$, then the CWE construction in Figure 14 is a CPA secure CWE scheme.*

Theorem 7 (CPA Secure CWE from CDH)**.** *Let Σ be a certificate-extractable signature scheme w.r.t. a one-way function OWF. If the CDH assumption is hard in $\mathbb{G}$ w.r.t. OWF, and H modeled as a random oracle, then the CWE construction in Figure 14 is a CPA secure CWE scheme.*

Lemma 10. *The CWE scheme in Construction 4 is correct.*

Proof. To verify correctness of Construction 4, we have to check, that for all messages m, m', verification keys vk, and certificate pairs $C \leftarrow \mathsf{PubC}(\mathsf{vk}, R, m)$, $c \leftarrow \mathsf{PrivC}(\sigma)$, where σ is a valid signature w.r.t. (m', R, vk) it holds that $m = \mathsf{Dec}_{\mathsf{CWE}}(c, \mathsf{Enc}_{\mathsf{CWE}}(C, m))$. By the certificate extractability of the signature scheme, we have that $C = c$; hence, we have

$$\frac{c_2}{c_1^c} = \frac{\mathsf{H}([c \cdot y]) \cdot m}{\mathsf{H}([c \cdot y])} = m.$$

Henceforth, Construction 4 is correct. □

Proof of Theorem 6. This proof primarily follows the proof of Theorem 2, with the main difference being the need to incorporate DH simulatability and certificate-extractability as abstract concepts, both provided by Schnorr signatures. Assume, for contradiction, that there exists an adversary $\mathcal{A}$ that can break the CPA security of Π_{CWE} with a non-negligible probability. We will use this adversary to build a reduction from the CPA security of Π_{CWE} to the hardness of DDH. The reduction $\mathcal{R}$ receives the values $([\alpha], [\beta], [\gamma])$, where γ is either a random value or $\alpha \cdot \beta$. The reduction runs $\mathsf{SimDDH}([\beta])$ to obtain the values $(\mathsf{vk}, R, x_0, x_1)$, such that $C = x_0[\beta] + x_1$. The reduction then forwards the key vk to $\mathcal{A}$, which eventually outputs three messages (m_0, m_1, m). The reduction computes the challenge ciphertext for the challenge message m_b as follows:

$$c_b \leftarrow ([\alpha], \mathsf{H}(x_0 \cdot [\gamma] + x_1[\alpha]) \cdot m_b).$$

Note that in Theorem 6, the hash function is the identity, so this computation matches the construction. Furthermore, by the DH simulatability, the value $x_0 \cdot [\gamma] + x_1[\alpha]$ equals $[\alpha \cdot c]$. The reduction forwards c_b to $\mathcal{A}$, which eventually outputs a bit b', and $\mathcal{R}$ returns $b' == b$.

Claim 6. *If $\gamma \neq \alpha \cdot \beta$, then $\mathcal{A}$ wins with probability at most $1/2$.*

Proof. If $\gamma \neq \alpha \cdot \beta$, then γ is a random element in $\mathbb{Z}_p$. Thus, the challenge ciphertext c_b is also a random element in $\mathbb{Z}_p$, since $x_0 \neq 0$ and is independent of the challenge message m_b. So, the claim follows. □

Claim 7. *If $\gamma = \alpha \cdot \beta$, then $\mathcal{A}$ wins with non-negligible advantage ε.*

Proof. If $\gamma = \alpha \cdot \beta$, then the ciphertext is exactly constructed as the adversary expects it to be. As we assume by contradiction that $\mathcal{A}$ breaks the CPA security of the encryption scheme with non-negligible probability, this probability carries over if $\gamma = \alpha \cdot \beta$. □

With the standard argument for the CPA security of the original Elgamal encryption, it holds that

$$\begin{aligned} \left|\Pr\left[\mathsf{DDH}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{DDH}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \\ \left|\Pr\left[\mathsf{CPA}^1_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{CPA}^0_{\mathcal{A}}(\lambda) = 1\right]\right| \geq \\ \left|\frac{1}{2} + \varepsilon - \frac{1}{2}\right| \geq \varepsilon. \end{aligned}$$

This contradicts the hardness of DDH; thus, no efficient adversary against the CPA security of Construction 4 can exist. □

Proof of Theorem 7. To prove Theorem 7, we assume via contradiction, that there exists an adversary against the CPA security of Construction 4 and build a reduction that leverages the capabilities of this adversary to compute a solution for a CDH instance. As input, the reduction gets a CDH-pair $[\alpha], [\beta]$. Following the proof of Theorem 6, the reduction uses the SimDDH algorithm to obtain values $(\mathsf{vk}, R, x_0, x_1)$, such that $C = x_0[\beta] + x_1$. The reduction then forwards the key vk to $\mathcal{A}$, which eventually outputs three messages (m_0, m_1, m). The reduction computes the challenge ciphertext for the challenge message m_b as follows by sampling a random value $h \leftarrow\!\!\$\ \mathbb{G}$ and outputting

$$c_b \leftarrow ([\alpha], h \cdot m_b).$$

Since the value h is randomly sampled and the RO models a truly random function, the adversary can only distinguish the message encrypted in the challenge ciphertext with a higher probability than guessing if it queries $[\alpha \cdot c] = [\alpha(x_0\beta + x1)] = x_0[\alpha\beta] + x_1[\alpha]$ at least once to the random oracle. To catch the respective query, the reduction guesses the occurrence of this query amongst all random oracle calls and aborts the simulation by outputting the element

$$\frac{h_1 - x_1[\alpha]}{x_0},$$

where h_1 is the respective query. If the guess is correct, the simulator successfully returns $[\alpha \cdot \beta]$, since the DH simulatability holds. Hence, the simulator outputs a valid solution for the CDH problem. The probability that the guess is correct is polynomially bounded since the adversary can only make polynomially many RO queries. Thus, there cannot be an efficient adversary against the CPA security of Construction 4. □

Instances. We now demonstrate that the BBS [BBS04] signature scheme complies with Theorem 7. Thus, we have CPA-secure CWE encryptions for BBS+. We defer the examples of the other signature schemes to Appendix B.3. We show in this section that the BBS+ signature is a certificate-extractable signature scheme. Since the BBS+ is defined w.r.t. a group capable of bilinear pairings, we show that the BBS+ signature satisfies the requirements of Theorem 7, i.e., BBS+ is DH simulatable and lives in a CDH-hard group.

Lemma 11 (BBS+ is CE)**.** *The BBS+ signature scheme is a certificate-extractable DH simulatable signature scheme.*

Proof. A BBS+ signature has the form (A, e, r), where $r \leftarrow\!\!\$\ \mathbb{Z}_p$. We define R as g_1^r where g_1 is part of the verification key. To show certifiability, we define $\mathsf{PubC}(\mathsf{vk}, m, R) := R$ and $\mathrm{PRIV}(A, e, r) := r$. Having available a signature (A, e, r) and a verification key vk, we define $\mathsf{ExtR}(\mathsf{vk}, (A, e, r)) := g_1^r$, which is well-defined, since g_1 is part of the verification key. Moreover, it holds that $R = [r]$, so one-wayness holds. So, BBS+ is a certificate-extractable signature scheme. To show DH simulatability, we define SimDDH via $\mathsf{SimDDH}([\beta]) := (\mathsf{vk} \leftarrow\!\!\$\ \mathbb{G}, [\beta], [1], [0])$.This way, for all m in the message space, $[\alpha \cdot c]$ can truly be expressed via $[\alpha \cdot c] = [\alpha \cdot r] = [\alpha \cdot \beta]$. □

Since the security of the BBS+ signature scheme is based on q-SDH [BBS04], which follows from CDH, all the assumptions of Theorem 7 are satisfied, and Construction 4 is a CPA secure CWE scheme for BBS+.

D.1 Certificate Extractable Signature Schemes

In this section, we show that Schnorr [Sch91], the Camenisch-Lysyanskaya (CL) [CL03], (Waters+) [BSW06], and Katz-Wang [KW03] signature schemes are certificate extractable signature schemes which are DH simulatable.

CL Signatures. CL signatures are defined w.r.t. the quadratic residues modulo n QR_n. A public key of a CL signature has the form $\mathsf{vk} = a, b, c, n$, and a signature of the CL signature scheme consists of the elements (v, e, r), where $v^e \equiv a^m \cdot b^r \cdot c \mod n$.

Lemma 12. *The CL signature scheme [CL03] is a certificate-extractable DH simulatable signature scheme.*

Proof. We define $R := b^r$, which is extractable from a signature (v, e, r). We define $\mathsf{PubC}(\mathsf{vk}, m, R) := R$ and $\mathsf{PrivC}(v, e, r) := r$, such that it holds $C = R = [r] = [c]$. To show DH-simulatability, we set $\mathsf{SimDDH}([\beta]) = (\mathsf{vk}, [\beta], 1, 0)$, where vk is a normally crafted encryption key, and b has the same basis as β. □

Waters+ Signatures. The Waters+ signature arises by the waters signature if one applies the BSW transformation [BSW06] to the waters signature [Wat05]. We defer a formal description of the BSW compiler to [BSW06], since we only rely on the fact, that BSW compiled signatures have the randomness in plain, and use a commitment to this randomness in the signature.

Lemma 13. *The Waters+ signature scheme [BSW06] is a certificate-extractable DH simulatable signature scheme.*

Proof. The structure of CL signatures and BSW compiled signatures is similar, i.e. the signatures consist of three elements (σ_1, σ_2, r), where a random commitment $[r]$ is used to compute σ_1. Hence, we follow the proof of CL and set $C = R = [r]$ and $c = r$, which shows all needed properties. □

Katz-Wang Signatures. A signature of the Katz-Wang signature scheme [KW03] consists of the three elements (A, B, z), for some $r \leftarrow\!\!\$\ \mathbb{Z}_p$, and random group elements $g, h \in \mathbb{G} \subset \mathsf{vk}$, $A = g^r, B = h^r$, and $z = \mathsf{H}(\mathsf{vk}, A, B, m) \cdot \mathsf{sk} + r$. The security of the Katz-Wang signature scheme is based on the hardness of DDH in $\mathbb{G}$.

Lemma 14. *The Katz-Wang signature scheme [KW03] is a certificate-extractable DH simulatable signature scheme.*

Proof. To show the certificate-extractability, we define $R := A$, $\mathsf{PubC}(\mathsf{vk}, R, m) := A \cdot \mathsf{vk}^{\mathsf{H}(\mathsf{vk}, A, B, m)}$, and $\mathsf{PrivC}(A, B, z) := z$. It holds, that $C = [z] = [c]$. Moreover, $R = A$ can be recovered from a signature σ. The Katz-Wang signature scheme is DH simulatable when defining $\mathsf{SimDDH}([\beta]) = ([\beta], [r], h, r)$ for a random $r \leftarrow\!\!\$\ \mathbb{Z}_p$, and a random h. For DDH simulatability to hold, we have to program the RO such that $\mathsf{H}(\mathsf{vk}, A, B, m) = h$. □

Schnorr Signatures. We have already introduced Schnorr signatures since a combined Sparkle signature verifies as a Schnorr signature. Moreover, we have implicitly shown in our warm-up section that Schnorr signatures are certificate-extractable and DH-simulatable. For a more formal treatment, we refer to the paragraph of the Katz-Wang signatures, which imply the DH simulatablity and certificate-extractrability of Schnorr.

E Impossibility Result

In this section,we formally prove our impossibility result. We begin by introducing auxiliary lemmas and definitions.

Lemma 15 (Extractability)**.** *Let Π_{MPFE} be a multi-party fair-exchange protocol of a coin for a witness w that achieves buyer fairness. Let $\mathcal{C}^*$ be the blockchain state, in which the first signature σ_{B} signed by the buyers w.r.t. a transaction from the buyers is contained and yields into a valid payment from the buyers. This blockchain state allows each buyer to locally extract the witness w.*

Proof. We prove this lemma using proof via contradiction and show how the buyer fairness breaks for Π_{MPFE} if the lemma does not hold. We assume, via contradiction, that $\mathcal{C}^*$ contains a valid payment from the buyer, but the buyer cannot learn the witness using its internal state and $\mathcal{C}^*$. If the buyer cannot extract the witness using $\mathcal{C}^*$, we define the following adversary: The adversary interacts with the buyer in the protocol and corrupts all sellers. At this point, where the signature σ_{B} is posted on the chain, the adversary aborts all communication and aborts the protocol. Then, the buyer cannot extract the witness, since there is no further message from any seller. In addition, the buyer paid the sellers some amount, since this is the signature σ_{B}. Therefore, our adversary breaks buyer fairness. □

Definition 25 (Final Chain)**.** *We call the blockchain state of Lemma 15 that allows the buyers to extract the witness* final blockchain state *$\mathcal{C}^*$.*

Lemma 16 (Last Message)**.** *Let Π_{MPFE} be a multi-party fair-exchange protocol of a coin for a witness w that achieves seller fairness. There exists a message m^*, which we call* final message*. When a party receives the final message, then this party is able to compute and post the final*

signature σ_{B} conveying the blockchain into the final blockchain state $\mathcal{C}^*$. The first final message is received by a seller, and before this seller receives the final message, no party can compute the signature σ_{B}.

Proof. We show, that this lemma must hold if seller fairness holds. We assume a single corrupted buyer for simplicity. The last message must exist since otherwise the malicious buyer could compute this signature on its own but not post it. This implies by Lemma 15, that the malicious buyer can use the signature σ_{B} and the blockchain state right before posting σ_{B} to extract the witness w locally. This breaks seller fairness, since the buyer does not pay any seller and yet learns the witness w. Hence, such a last message must exist. In addition, this implies that the recipient of the final message must be a seller. Otherwise, the above attack again breaks seller fairness.

□

Proof of Theorem 4. We prove this theorem by providing an adversary that breaks seller fairness with non-negligible probability if it can corrupt $t_{\mathsf{S}} - 1$ sellers (and also all buyers by definition of seller fairness).

To start the attack, the adversary corrupts all buyers and a random majority of sellers. Eventually, the last message is sent to one seller (c.f. Lemma 16). If the receiving seller is not a corrupted one, the adversary aborts. This happens with probability $\frac{n_{\mathsf{S}}-t_{\mathsf{S}}+1}{n_{\mathsf{S}}}$. If the adversary does not abort, the adversary learns the final message in this round and can compute the final signature (by definition of the last message). Knowing both the final signature and the internal state of each buyer (this is the case since the adversary controls all buyers), the adversary can extract the witness locally (Lemma 15). To end the round, the adversary invalidates each unspent payment of the buyer by computing a valid signature w.r.t. the buyer's signing key (since the corrupted parties share the state, this is possible) and all possible input transactions controlled by the buyer and posts this signature on the blockchain invalidating all open input transactions (especially the inputs of the final signature). After extracting the witness and invalidating each possible payment from the buyer (the buyer did not post a valid signature on the chain yet since this would be the final signature, so the buyer still controls all initial input transactions), we have two cases:

1. In the first case, all honest sellers are paid their share of the coin.
2. In the second case, at least one honest seller does not receive a payment with its share of the coin. This violates seller fairness, as the adversary has extracted a valid witness while not compensating all honest sellers.

In the following, we show that the case in which the adversary does not break seller fairness (Case 1) only happens with negligible probability.

Claim 8. *The probability that all honest sellers are paid their share of the coin is negligible.*

Proof. For this claim, we are in the setting that there is a seller, S^*, that received the last message but did not post a signature on-chain, which would have triggered a payment of the buyer. At this point, posting such a signature has become impossible since the buyer has already invalidated all its input transactions. In this case, the protocol has to distinguish whether S^* is corrupted or not to attain fairness: If S^* is corrupted, this seller can collude with the corrupted buyers and learn the

witness (as described above). Then, the remaining sellers would have to spend $n_{\mathsf{S}} - 1/n_{\mathsf{S}}$ coins held by S^* to pay all honest sellers. Otherwise, seller fairness breaks, since the adversary can output a valid witness while at least a single honest seller is not paid. If S^* is not corrupted, the remaining sellers must not spend the coins of S^*, since otherwise seller fairness breaks. (The honest S^* would have fewer funds after the protocol than before.) To prove this claim, we show that the remaining (honest) protocol participants cannot distinguish whether the witness leaking S^* is corrupted or not. To prove this indistinguishability, we recall the actions of a witness leaking S^*: S^* follows the protocol honestly, receives the final message, and continues to follow the protocol honestly. When S^* finishes the round, the adversary that controls the network implants the invalidation signature on-chain, such that even when following the protocol honestly, the final signature does not refer to a valid transaction on-chain. Since the corrupted S^* did not deviate from executing the protocol honestly (despite sharing its internal state with $\mathcal{A}$, which is not detectable by all honest parties), the actions of this corrupted S^* cannot be distinguished from the actions of an honest S. Thus, the claim holds.

□

Using this claim, we can deduce that the adversary breaks seller fairness with overwhelming probability if it receives the last message. Still, this happens with non-negligible probability $\frac{n_{\mathsf{S}} - t_{\mathsf{S}} + 1}{n_{\mathsf{S}}}$. Therefore, we have shown that no multi-party fair exchange protocol Π_{MPFE} that achieves t_{B}-buyer fairness can also achieve t_{S}-seller fairness in the presence of only a minimal blockchain. This closes the proof of Theorem 4.

□

Impossibility with Timelocks and Dishonest Majority. Now, we show the impossibility of MPFE protocols for minimal blockchains with timelocks and a dishonest majority of sellers. To this extent, we update our setting, such that we have a minimal blockchain with timing. In addition, in our adversarial model, we assume that the adversary can delay messages at most $t_{\mathcal{A}}$ rounds and assume that the time of a timelock is a multiple of the maximal computation time of a round. Therefore, we can assume that each party can run computations while coins are locked, and the adversary cannot circumvent the locking by delaying messages. In this setting, we show our impossibility. We start with an auxiliary lemma.

Lemma 17. *If less than t_{S} honest sellers participate, there exists an adversary for every* MPFE *protocol, such that only the adversary can receive any last message.*

Proof. In the setting of seller fairness, the adversary controls $t_{\mathsf{S}} - 1$ sellers and all buyers. To compute the final signature, shares $s_1, \ldots, s_{n_{\mathsf{S}}+n_{\mathsf{B}}}$, based on the secret inputs of the parties, are required. Any party can compute the final signature locally if it knows enough of these shares. Using our reflective thresholds t_{S} and t_{B}, this party needs the shares of t_{B} buyers and the shares of t_{S} sellers. The adversary controls all buyers and $t_{\mathsf{S}} - 1$ sellers. Hence, the adversary only needs the share of a single honest party to have enough shares available to compute the final signature. The honest parties alone lack enough shares since at most $t_{\mathsf{S}} - 1$ honest sellers participate in the protocol. We run our protocol sequentially in rounds, and we can assume that, eventually, the participants reveal their shares to the other participants. In particular, there must be a seller S^* revealing its share first to another party. Since S^* reveals its share first, the share of no other seller

is known to any other party at this point. We define the following adversary $\mathcal{A}$ that guarantees that no honest party ever learns a final message: $\mathcal{A}$ guesses the identity of S^* and leaves S^* uncorrupted. In addition, $\mathcal{A}$ guesses the receiver of the first exchanged share and corrupts this receiver. Since the receiver is corrupted, $\mathcal{A}$ knows the shares of all buyers and t_{S} shares of the sellers. Hence, $\mathcal{A}$ can compute the final signature (thus, this message is the final message for $\mathcal{A}$). Since the seller who sends this message is honest, this message cannot be the last message for honest sellers since, at this time, no seller has shared its share yet, and hence honest parties know at most $t_{\mathsf{S}} - 1$ many shares. In addition, no other seller has revealed its share yet, since S^* was the first one. No dishonest seller will send its share after the first final message is received. Therefore, no honest seller can learn the signature, and thus no honest seller learns any last message. We want to emphasize that our result is unaffected by timed cryptography, since even with timed commitments, the PPT adversary can eventually learn the committed message. □

Proof of Theorem 5. We use Lemma 17 to build an adversary that learns the final message, while no honest party receives any final message. Using this adversary we build the following adversary that breaks seller fairness: $\mathcal{A}$ learns the final message and computes the final signature. $\mathcal{A}$ breaks seller fairness if there is an honest seller who did not receive their share of the coin. We already know, that there is no payment from the buyers to the sellers since this would be associated with the final signature that is not posted yet. Therefore, there must be a payment from the dishonest sellers to the honest sellers to attain seller fairness. In the following, we show, that this cannot be the case for corrupted buyers and a dishonest majority of sellers. We have two cases: In the first case, all honest sellers are paid *before* the adversary receives the final message. In the second case, all honest sellers are paid *after* the adversary receives the final message. In the following, we show that the probability for both cases is negligible.

Claim 9. *The probability that all honest sellers are paid* before *the adversary receives the final message is negligible.*

Proof. Suppose the honest sellers have already been paid before the final message has been sent. In that case, we know that this payment cannot come from the buyers (otherwise, the signature for this payment would be the final signature and the final message would have been exchanged already). Thus, there exists a seller whose balance after this payment is higher than before the payment. We call this seller S^*. We define the following adversary that breaks seller fairness: The adversary corrupts all buyers and guesses which seller is S^* and corrupts this seller and a set of $t_{\mathsf{S}} - 2$ other sellers (if $t_{\mathsf{S}} > 2$), such that there remains at least a single honest seller, who pays S^*. Once the seller S^* is paid, the adversary terminates the protocol but forwards messages of honest parties. This adversary breaks seller fairness if the guess of S^* is correct since S^* is dishonest and has more funds after the protocol than before (by our case assumption). In addition, the adversary refuses communication of all dishonest parties before a final share is exchanged, and hence no party can receive a last message. Since coins also cannot be crafted by the sellers, there must be at least a single honest seller who has fewer funds after the protocol than before. This adversary would break seller fairness, and hence, there is at least a single honest seller that was not paid before the adversary received the final message. □

Claim 10. *The probability that all honest sellers are paid* after *the adversary receives the final message is negligible.*

Proof. If all honest sellers are paid after the adversary receives the final message and stops interacting in the protocol (while still forwarding messages of the honest parties), this payment is made by the honest sellers in one of two ways: Either the minority of $t_{\mathsf{S}} - 1$ honest sellers can enforce this payment on their own (we call this case the minority payment case), or the minority of ($t_{\mathsf{S}} - 1$ honest) sellers already received shares of the adversary before the final message was delivered, which now allows them to make the payment (we call this case the majority payment case). To show that the claim holds, we show that the probability of each case to happen is negligible.

To show the negligible probability for the minority payment case, we observe that if a minority can enforce a payment that spends funds of a party that is not part of this minority, the adversary can simply corrupt exactly this minority, and sufficiently more sellers such that at least a single honest sellers has less funds afterward, make the payment, and terminate the protocol. This breaks seller fairness since at least a single seller has less balance at the end of the protocol than before. More formally, we define the following adversary against seller fairness: The adversary corrupts all buyers, guesses the minority that can trigger the payment, and corrupts exactly this minority. In addition, the adversary also guesses which seller will make the payment (i.e., will have fewer funds after the payment) and corrupts sufficiently many more sellers, such that the one spending party remains honest. The adversary uses the power of the minority and makes a payment that spends coins of at least a single seller which is honest. Then, the adversary terminates the protocol but continues forwarding messages of honest parties. This adversary breaks seller fairness since no payment of the buyer can be triggered in this corruption scenario: The adversary did not send the share of a malicious seller, so no honest seller can receive the final message.

Next, we show the negligible probability for the majority payment case. If a minority of sellers receives sufficient information to punish another seller before this seller learns the final message, we can flip corruption to violate seller fairness: The adversary corrupts the minority that triggers the payment and does not corrupt the party that will be punished. In addition, $\mathcal{A}$ corrupts sufficiently many more sellers. Once the party to be punished reveals the information needed for punishment (this happens before the final message is sent), the adversary triggers the punishment and terminates the protocol. Therefore, an honest seller is punished. The adversary breaks seller fairness if no payment from the buyers to the sellers appears on-chain. This is indeed the case since the final message was not sent when the adversary punished the first receiving seller. In addition, the adversary does not send any messages after punishing and the final message can only be sent by the adversary (dishonest majority of sellers). □

Taking together both claims, we can deduce that the adversary breaks seller fairness with overwhelming probability if it receives the last message. This holds since $\mathcal{A}$ learns the witness, but there exists an honest seller that is not compensated for revealing its partial witness.

□

F Additional Figures

In this section, we provide additional figures.

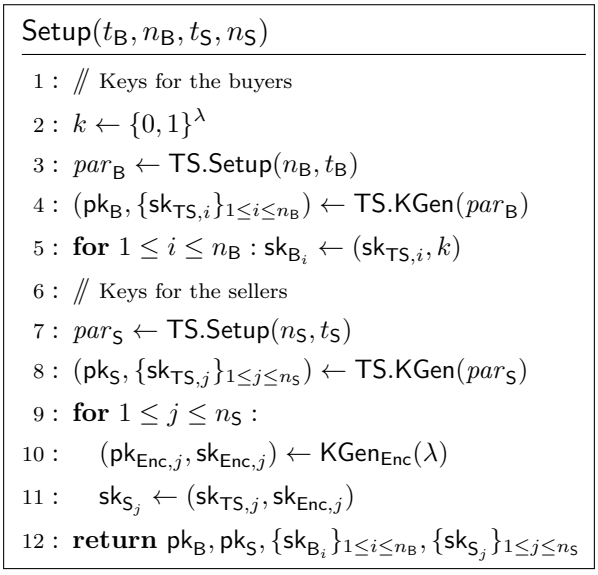

Figure 15: The setup algorithm.