Foundations of Adaptor Signatures

Paul Gerhart[1], Dominique Schröder[1,4], Pratik Soni[2], and Sri AravindaKrishnan Thyagarajan[3]

[1] *TU Wien*
[2] *University of Utah*
[3] *University of Sydney, Sydney, Australia*
[4] *Friedrich-Alexander-Universität Erlangen-Nürnberg*

Abstract

Adaptor signatures extend the functionality of regular signatures through the computation of *pre-signatures* on messages for statements of NP relations. Pre-signatures are publicly verifiable; they simultaneously hide and commit to a signature of an underlying signature scheme on that message. Anybody possessing a corresponding witness for the statement can adapt the pre-signature to obtain the "regular" signature. Adaptor signatures have found numerous applications for conditional payments in blockchain systems, like payment channels (CCS'20, CCS'21), private coin mixing (CCS'22, SP'23), and oracle-based payments (NDSS'23). In our work, we revisit the state of the security of adaptor signatures and their constructions. In particular, our two main contributions are:

- **Security Gaps and Definitions:** We review the widely-used security model of adaptor signatures due to Aumayr *et al.* (ASIACRYPT'21) and identify gaps in their definitions that render known protocols for private coin-mixing and oracle-based payments *insecure*. We give simple counterexamples of adaptor signatures that are secure w.r.t. their definitions but result in insecure instantiations of these protocols. To fill these gaps, we identify a minimal set of modular definitions that align with these practical applications.
- **Secure Constructions:** Despite their popularity, *all* known constructions are (1) derived from identification schemes via the Fiat-Shamir transform in the random oracle model or (2) require modifications to the underlying signature verification algorithm, thus making the construction useless in the setting of cryptocurrencies. More concerningly, *all* known constructions were proven secure w.r.t. the insufficient definitions of Aumayr et al., leaving us with no provably secure adaptor signature scheme to use in applications.

  Firstly, in this work, we salvage *all* current applications by proving the security of the widely-used Schnorr adaptor signatures under our proposed definitions. We then provide several new constructions, including presenting the *first* adaptor signature schemes for Camenisch-Lysyanskaya (CL), Boneh-Boyen-Shacham (BBS+), and Waters signatures, all of which are proven secure *in the standard model.* Our new constructions rely on a new abstraction of digital signatures, called *dichotomic signatures*, which covers the essential properties we need to build adaptor signatures. Proving the security of all constructions (including identification-based schemes) relies on a *novel non-black-box* proof technique. Both our digital signature abstraction and the proof technique could be of independent interest to the community.

Contents

1 Introduction

Adaptor signatures allow a signer to compute a verifiable promise w.r.t. some NP statement Y, called a pre-signature. The signer promises that anyone who knows a witness y for Y can compute a correct signature from the pre-signature and otherwise cannot. Adaptor signatures are extractable; given a pre-signature and a signature, one can efficiently extract the witness y. Adaptor signatures have numerous applications in the blockchain space, such as payment channels [DW15; Mil+19; Eck+20], private coin mixing [Gla+22; Qin+23], and oracle-based payments [Mad+23], among many others.

Security Definitions. Adaptor signatures were initially defined for the one-time fair exchange of a coin for a witness; now, they serve in various settings that deviate from this two-party fair exchange as outlined above. Given that the security definitions were initially formulated exclusively for one-time fair exchange scenarios, the question arises as to whether the security model has the necessary strength for new applications. Recently, Dai *et al.* [DOY22] revisited the security model and proposed stronger definitions. The authors justify the need for one of their definitions with a contrived counter-example identifying a definitional gap. Unfortunately, this definitional gap remains purely theoretical and is not applicable to novel applications, which brings us back to our original question:

- *Which definitions are both necessary and sufficient for real adaptor signature applications?*

We re-examine the security of adaptor signatures in the context of the new applications and identify multiple vulnerabilities that lead to specific attacks within each application. These attacks have two implications. First, the security proofs of nearly all recent applications have gaps. We illustrate these gaps by creating artificial schemes that maintain security according to the original definitions [Erw+21; Aum+21]. However, when our scheme is used in private coin mixing [Gla+22; Qin+23] or the oracle-based payments protocol [Mad+23], it leads to insecure protocols. Our attacks confirm the necessity for stronger definitions as previously suggested by Dai et al. [DOY22]. However, our attacks also reveal that a very basic property is missing to securely realize these applications. We close this gap by introducing *pre-verify soundness*, a property that ensures that the pre-verification algorithm satisfies (computational) soundness w.r.t. the NP relation Rel. Second, our attacks expose weaknesses in the original security proofs within all practical applications of adaptor signatures, such as private coin mixing [Gla+22; Qin+23] and oracle-based payment [Mad+23] protocols. However, specific implementations that rely on the Schnorr adaptor signature within the random oracle framework remain unaffected (we prove the security of Schnorr adaptor signatures w.r.t. the stronger definitions within our general framework). This observation underscores that practitioners, when designing innovative applications, prioritize a distinct set of security properties that may not be generically covered by current adaptor signature formalization. Given the rapid development and integration of adaptor signatures in the blockchain domain, there is an urgent need to bridge the gap between intuitively understood security properties and the underlying formal guarantees. To address this, we strive to narrow this gap by extracting a minimal set of modular definitions that align with these practical applications.

Constructions. Adaptor signatures are always defined w.r.t. some underlying *fixed* signature scheme. The underlying signature scheme is typically hardcoded in some blockchain application prohibiting any modification to the scheme itself, especially not the verification algorithm. In other words, any adaptor signature scheme must end up issuing signatures that are verified under the original verification algorithm of the signature scheme. This practical limitation makes the construction of adaptor signatures challenging. It leads to the fact that all known constructions are based on identification schemes that are converted to a signature scheme using the Fiat-Shamir transformation in the Random oracle model [BR93; EEE20; TMM20; Aum+21; Erw+21; Alb+22]. The only exception is the work of Aumayr et al. [Aum+21], where they construct an adaptor signature scheme for ECDSA based on concepts from [Mal+19]. Unfortunately, the security of *all* known schemes is proven secure only in the model proposed by Aumayr *et al.* [Aum+21], which we show is insufficient for novel applications. This leads to an unsatisfactory situation in which practitioners use unproven constructions, and all possible candidate schemes achieve security only within the random oracle model.

Relying on the random oracle model is undesirable for at least two reasons. First, the random oracle model is not sound, as demonstrated by Canetti, Goldreich, and Halevi [CGH04]. Consequently, the pursuit of constructions within the standard model is crucial to instill greater confidence in the existence of this cryptographic primitive in general. Second, using the Fiat-Shamir transformation causes the signature algorithm not to work well in all (practical) applications, such as anonymous credentials, which hinders further adaptation of this primitive. The challenge here is the application of the hash function, which often cannot be (efficiently) integrated into complicated zero-knowledge proofs. Therefore, we put forward the following questions:

- *Does the Schnorr adaptor signature scheme satisfy the stronger security notions?*
- *Given a standard model signature scheme Σ, can we construct a standard model adaptor signature scheme* AS *based on Σ without modifying Σ?*

We answer both questions in the affirmative. In particular, we present the first adaptor signatures that do not rely on the random oracle heuristic. Instead of proving security for all schemes from scratch, we develop a general framework, referred to as *dichotomic adaptor signatures*, that covers all known ID-based adaptor signatures and several standard model signature schemes, such as CL signatures [CL03], the strongly unforgeable Waters signature scheme [BSW06], and the BBS+ [BBS04] signature scheme, which is currently being standardized. When instantiating our framework with the strongly unforgeable Waters signature scheme [BSW06], we obtain the first adaptor signature scheme that is secure under the CDH assumption, one of the weakest cryptographic assumptions used in practice. The security proof includes a novel non-black-box proof technique called *transparent reduction*, which may also be of independent interest to the community.

1.1 Our Contribution

Our contribution can be summarized as follows.

- We show that the initial formal security model for adaptor signatures leads to insecure instances when adaptor signatures are applied in real-world scenarios. To ensure the safe and reliable use of adaptor signatures in practical applications, we extract and simplify the core concepts of the stronger definitions by Dai, Okamoto, and Yamamoto [DOY22] and introduce a missing security property required by the real-world applications, called pre-verify soundness.

- We present the first adaptor signature schemes in the standard model without modifying the signature verification algorithms of the underlying signature schemes; a crucial requirement for practical applications which rely on standardized primitives. Concretely, we design adaptor signature schemes based on the signature schemes of Camenisch and Lysyanskaya (CL) [CL03], Boneh, Boyen, and Shacham (BBS+) [BBS04] that are widely used in modern digital systems in the context of Anonymous Credentials. Furthermore, we also present an adaptor signature scheme based on the strongly unforgeable Waters signature (Waters+) [BSW06].

- Our third contribution is conceptual. We analyze the algebraic properties of the underlying signature schemes of all known adaptor signatures to find a common blueprint that can then be instantiated in the standard model. We call our resulting abstraction *dichotomic signatures* and show that it covers all known schemes (that lead to current adaptor constructions) as well as the CL, BBS+, and Waters+ signature schemes. Our adaptor signature constructions are highly efficient and add little to no overhead over the algorithms of the underlying signature schemes.

- On a technical level, we present a novel non-black-box proof technique called *transparent reductions*, which exploits the code of a reduction from the signature scheme to the underlying hard problem in a non-black-box way. This technique allows us to formally analyze the security of our adaptor signature schemes based on dichotomic signatures in the standard model, i.e., not relying on the ROM. Since CL, BBS+, and Waters+ are signature schemes in the standard model, this gives us the first adaptor signature schemes in the standard model (provided that the signature verification algorithms are not modified). In addition, we show that transparent reductions can also be used to prove the security of Schnorr adaptor signatures. We believe that transparent reductions are indeed necessary to prove standard model adaptor signatures secure: Without the powerful toolset of the random oracle model (namely programming and witness extractable NIZKs), it seems impossible to simulate a pre-signature oracle without violating the strong unforgeability of the underlying signature scheme.

1.2 Related Work

Adaptor signatures made their debut in the work of Poelstra [Poe17] and were later formally defined by Aumayr et al. [Aum+21]. They have versatile applications, spanning various domains such as Payment Channel Networks (PCNs) [Mal+19; DW15; Mil+19; Eck+20], private coin mixing [Gla+22; Qin+23], and Oracle-based payments [Mad+23], among others. In the realm of post-quantum cryptography, Esgin *et al.* [EEE20] and Tairi *et al.* [TMM20] contributed secure adaptor signatures within the random oracle model. The construction of Esgin *et al.*, a post-quantum adaptation of Schnorr, falls under the category of dichotomic signatures. The concept of adaptor schemes for a class of signature schemes, particularly those secure in the random oracle model, saw its initial generalization by Erwig *et al.* [Erw+21]. Our abstract representation of dichotomic signatures encompasses all known schemes and includes innovative standard model constructions. Verifiably encrypted signatures, akin to adaptor signatures, differ in their inability to support the public adaptation of the signature [Bon+03; RS09] or the extraction of the witness from the signature and ciphertext.

A recent contribution by Dai et al. [DOY22] introduced new definitions for adaptor signatures and an adaptor signature scheme that any unforgeable signature scheme can instantiate. While we believe the stronger security definitions to be adequate, we extract a necessary, simplified core of these definitions. Moreover, their standard model adaptor signature construction necessitates modifications to the underlying signature verification algorithm, circumventing a fundamental challenge in adaptor signature design. This adjustment at the application layer has ramifications, as it compromises the fungibility of blockchain transactions and renders them incompatible with many existing blockchain systems.

Tairi, Moreno-Sanchez, and Schneidewind took a distinct approach in their quest to establish robust security definitions for adaptor signatures [TMSS23]. They formulated security notions for adaptor signatures within the Universal Composability (UC) framework.

2 Technical Overview

In this section, we give an overview of our contributions towards improving *both* the foundations and the practice of adaptor signatures. Towards this, we begin with a description of the adaptor signature functionality in Section 2.1 and its formalization for the case of payment channels by Aumayr et al. [Aum+21]. Then, in Section 2.2, we give an overview of the gaps we identified in the existing security definitions of [Aum+21] along with explicit attacks that break the security of *three* higher-level protocols that use adaptor signatures. With these gaps in mind, we advocate the use of the recently introduced security definitions by Dai *et al.* [DOY22] for adaptor signatures. In Section 2.3, we switch focus and present our new general framework for constructing secure adaptor signatures. Finally, in Section 2.4, we discuss several instantiations of our framework, including presenting the *first* adaptor signatures for several signatures schemes, including Boneh-Boyen-Shacham (BBS+) [BBS04]. These instantiations result in the *first* adaptor signatures in the standard model.

2.1 Adaptor Signatures and Payment Channels

We begin by describing the adaptor signature scheme functionality, which was introduced by Poelstra [Poe17]. Informally, an adaptor signature scheme models a blockchain-based protocol for fairly exchanging a digital signature (e.g., on a transaction) and a witness for an NP statement. More specifically, the adaptor signature scheme consists of four algorithms ($\mathsf{pSign}, \mathsf{pVrfy}, \mathsf{Adapt}, \mathsf{Extract}$) which are used as follows for the fair exchange: a signer Alice uses the pre-signing algorithm pSign to compute a pre-signature $\widetilde{\sigma}$ on a transaction message m w.r.t. an NP statement Y held by Bob. Then, running the adapt algorithm Adapt on $\widetilde{\sigma}$ and its witness y, Bob computes a valid signature σ on the message m under Alice's signing key. To redeem the payment, Bob posts σ on the blockchain; this allows Alice to learn the witness y by running the extract algorithm $\mathsf{Extract}$ on the $(\widetilde{\sigma}, \sigma)$ pair. Thus, Alice obtains the witness for the statement Y, while Bob obtains the signature on the transaction message m.

The work of Aumayr *et al.* [Aum+21] showed a mechanism to bootstrap the above fair-exchange protocol to enable *payment channels*, a scalability solution for blockchain-based cryptocurrency payments. To formally capture security for this fair exchange, they proposed *three* essential security properties for adaptor signatures: *unforgeability*, *pre-signature adaptability*, and *witness-extractability.* For the sake of this overview, we will only focus on the property of witness-extractability, which is crucial for fairness. Informally, witness extractability ensures that Alice can successfully extract Bob's witness from *every* valid signature σ given by Bob. Without witness-extractability, there may exist a valid signature σ, which is sufficient for Bob to redeem the payment, but knowing σ may not allow Alice to learn the witness, thereby violating fairness. To formalize this, Aumayr et al. consider the following game and, for security, require that no efficient adversarial Bob wins the game with non-negligible probability:

- The challenger provides a signing and a pre-signing oracle to Bob.
- Eventually, Bob outputs as challenges a message m^* and a statement Y^*.
- The challenger computes a *single* pre-signature $\widetilde{\sigma}$ on (m^*, Y^*) and forwards $\widetilde{\sigma}$ to Bob, who again has access to the oracles and eventually outputs a signature forgery σ^*.
- Bob wins if he never asks any of the oracles on the message m^*, and the forgery does not extract with $\widetilde{\sigma}$, i.e., $(Y^*, \mathsf{Extract}(\widetilde{\sigma}, \sigma^*)) \notin \mathsf{Rel}$.

This formalization of witness extractability, where the adversary only has access to exactly one pre-signature on the challenge message m^*, is meaningful for the payment channel setup, where the messages on which the pre-signatures are computed on are unique transactions. Therefore, the adversary will never learn two pre-signatures on the same message, as each transaction is different and no transaction can be executed twice.

2.2 Gaps in Adaptor Signature Definitions

Our starting point is the observation that the above three properties are specifically tailored to the setting of payment channels. For example, for witness-extractability, it is sufficient that every valid signature on a message m computed by Bob is extractable when Bob has learned exactly one pre-signature on m before. Therefore, a Bob capable of generating a *non-extractable* signature after learning *two* pre-signatures on the same message m does not hurt payment channels. However, as we show next, such a Bob not only exists but also voids the security for adaptor-signature-based protocols beyond payment channels.

Beyond payment channels, adaptor signatures have found more broad applications, such as oracle-based payments [Mad+23], and (blind) coin-mixing services [Gla+22; Qin+23]. Despite being used in much broader contexts, the security is argued relying on the formalization of adaptor signatures tailored towards the case of payment channels. This mismatch allows us to break the security of all the above three protocols. In particular, for each of these protocols, we find adaptor signature schemes that (a) satisfy the security definitions in [Aum+21], but (b) render the higher-level protocol insecure. Henceforth, we refer to such adaptor signature schemes as *counter-examples*.

Counter-examples. The key idea towards designing counter-examples relies on exploiting the differences between the considered protocol's setting and that of payment channels. For example, for the case of oracle-based payments, we design an adaptor-signature with *leaky pre-signatures*: To build these, we use an adaptor signature scheme that is secure w.r.t. the definitions provided by [Aum+21]. To make a pre-signature on a message m leaky, we add a two-out-of-two share of a signature σ on m to the pre-signature. When a party only learns a single pre-signature, this two-out-of-two hides the additional signature σ. However, if one learns a second pre-signature on the same message, the combination of both shares can be used to extract σ. We show later in this overview that if instantiated with adaptor signatures with leaky pre-signatures, the security of oracle-based payments cannot be argued if a single transaction is pre-signed twice. However, computing a second pre-signature is a desirable goal for this application.

Similarly, for coin-mixing protocols [Gla+22] and blind-coin-mixing protocols [Qin+23], we design adaptor signature schemes where pre-signatures satisfy properties of *malleability* and *unadaptibility*. We postpone the description of these properties and the constructions to Section 4. We provide a summary of the protocols affected by our attacks in Figure 1 and continue with explaining how the class of leaky pre-signatures breaks the one-wayness of oracle-based payments.

Protocol		**Pre-Signatures in Our Counter-Examples**	
Coin Mixing	[Gla+22]	Malleability	(Section 4.1)
Oracle-Based Payments	[Mad+23]	Leaky	(Section 4.3)
Blind Hubs	[Qin+23]	Unadaptability	(Section 4.2)

Figure 1: An overview of which protocol is affected by which gap.

Insecurity of Oracle-Based Payments. Oracle-based payments allow two mistrusting parties, Alice and Bob, to make payments conditional on the outcome of some real-world event [Mad+23]. To do this, Alice sends verifiably encrypted signatures on transactions to Bob. These encrypted signatures can be decrypted by Bob when a threshold number of semi-trusted oracles sign predetermined messages. For example, if Alice wants to pay Bob 5 coins if the weather is rainy on a given date, she encrypts a signature on a transaction of 5 coins to Bob. If Bob receives the testimony of at least three weather oracles that it rained on the given date, he can decrypt the signature and redeem the payment. Adaptor signatures are used as the main building block for oracle-based payments. The encryption of a signature is a pre-signature that can be verified using the pre-verify algorithm. The pre-signature is generated w.r.t. a statement Y whose witness y is secretly shared among the oracles. When an oracle testifies for an event, it reveals its share of the witness y. We can see that it is natural for oracle-based payments to create multiple pre-signatures on the same message: If Alice wants to pay Bob 5 coins when it rains *or* when it snows, she creates a transaction for 5 coins and encrypts it once for rain and once for snow.

As we have already described, witness extractability only provides security guarantees if an adversary learns exactly one pre-signature on a challenge message m^*, since this correctly models the payment channels. Therefore, we can build an adaptor signature scheme in which any pair of pre-signatures on the same message reveals a full signature, even if the pre-signatures are not adapted. Such an adaptor signature scheme remains witness extractable (according to the definition from [Aum+21]) since the adversary, who still only learns one pre-signature on the challenge message, cannot obtain the additional valid signature on the challenge message. However, such an adaptor signature scheme defeats one-wayness, which is the most important security property of oracle-based payments: One-wayness guarantees that an encrypted payment for a threshold of t oracles (a pre-signature on a message m) can only be redeemed (learn a signature on m) if a malicious user provides the testimony of at least t oracles. Leaky pre-signatures break one-wayness since a malicious Bob asks Alice for two encrypted payments on the same transaction message m and learns a valid signature on m from this pair. Thus, Bob can spend transaction m without any testimony.

Stronger Definitions. Each of our counterexamples (c.f. Section 4) shows that the mentioned gaps in the original definitions of adaptor signatures are not theoretical but lead to security flaws in the broad applications of adaptor signatures. Thus, we try to find sufficiently strong definitions of adaptor signatures. The work of Dai *et al.* [DOY22] provides an important contribution to the security of adaptor signatures, as they model security without an implicit application of the primitive to payment channels. They also provide a separating counter-example between the new and the old definitions. However, the counter-example does not break the security of any published application that uses adaptor signatures. Therefore, it is unclear whether the new definitions are only theoretically stronger or if they indeed fix the gaps in the current adaptor security definitions. In contrast, we show that the gaps between the old and the new definitions are indeed practical, as all our explicit attacks are based on counter-examples that are secure w.r.t. the old definitions but insecure w.r.t. the new ones. The definitions of Dai *et al.* include *extractability, unique extractability,* and *unlinkability.* While these definitions cover some applications, we observe in Section 4.2, that there is one important definition missing, which we formalize as *pre-verify soundness.* Looking ahead, we will show in Section 5 that the three definitions of Dai *et al.* alongside pre-verify soundness are sufficient for current adaptor signature applications and we will stick to them: For example, if an adaptor signature scheme is *extractable*, then there are no leaky pre-signatures. The reason is that extractability allows an adversary to see multiple pre-signatures on the challenge message and thus allows an adversary to extract the additional provided information. We recite the stronger definitions and show how each counterexample is insecure w.r.t. the stronger security definitions in Section 5.

A Perspective. Dai *et al.* [DOY22] presented a contrived counterexample scheme to demonstrate the strength of their definitions. Since these counterexamples are not applicable to the practical applications of adaptor signatures, the cryptographic community continued to use the definitions of Aumayr et al. even after the work of Dai et al. was published, as evidenced by these publications [Gla+22; Qin+23]. In contrast, our (artificial) counterexamples are applicable to the practical applications of adaptor signatures. Therefore, we show that the community expects stronger security properties in their applications that are not currently provided.

2.3 A Framework for Constructing Adaptor Signatures

We now focus on our second main contribution, which is a new framework for constructing adaptor signatures. Currently, adaptor signatures are only known for ID-based signature schemes. Two significant challenges with ID-based adaptors are: (a) their security was proven under the definitions in [Aum+21] which, as demonstrated earlier, are insufficient beyond payment channels, (b) the only known approach to proving the security of ID-based signatures requires idealized models like the random oracle model. More concerningly, these security proofs use the "full" power of the random oracle model (i.e., programmability), and this is also inherited by the security proofs of the corresponding adaptor signatures. Whether the reliance on (the full power of) idealized models is necessary for proving security is a well-motivated fundamental question throughout cryptography. While the class of ID-based signatures is widely used in cryptocurrencies, there are several non-ID-based signature schemes like BBS+, Waters, and CL that boast desirable properties. Enabling adaptor signatures for them is interesting from both theoretical and practical perspectives.

Dichotomic Signatures. Towards providing a general framework that can capture several algebraic signature schemes, we identify three natural algebraic properties of signature schemes. We refer to the class of signatures as *dichotomic signatures.* These properties will allow us to build adaptor signatures for these signature schemes. A dichotomic signature scheme $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is a randomized signature scheme equipped with an injective one-way function $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$, where $\mathcal{D}_{\mathsf{R}}$ and $\mathcal{D}_{\mathsf{R}'}$ are randomness spaces mapping Σ's randomness such that the following properties are satisfied:

- *Decomposability:* there exists efficiently computable randomized algorithms Σ_1, Σ_2 such that for every signing key sk, message m, and randomness r, the signature $\sigma = \mathsf{Sign}(\mathsf{sk}, m; r)$ decomposes into a tuple (σ_1, σ_2) where σ_1 (resp., σ_2) uses $\mathsf{OWF}(r)$ (resp., r) as the random tape. That is,

$$\sigma_1 = \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r));\ \ \sigma_2 = \Sigma_2(\mathsf{sk}, m; r)\ .$$

- *Homomorphism:* Σ_2 admits an additive homomorphism w.r.t. its randomness component. That is, for every signing key sk, message m and randomness $r, y \in \mathcal{D}_{\mathsf{R}}$,

$$\Sigma_2(\mathsf{sk}, m; r + y) = \Sigma_2(sk, m; r) + y\ .$$

- *Verifiability:* a signature (σ_1, σ_2) can be verified by just checking an efficient relation between σ_1 and $\mathsf{OWF}(\sigma_2)$. This is captured by the existence of an efficient algorithm Vrfy' that satisfies

$$\mathsf{Vrfy}(\mathsf{vk}, m, (\sigma_1, \sigma_2)) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2))\ ,$$

for every verification key vk, and all messages m.

These properties are natural for algebraic signature schemes, and in fact, the abstraction of dichotomic signatures is quite expressive: In Section 6 we show several known schemes relying on different algebraic structures are dichotomic. Examples include the prime-order group based Schnorr signatures [Sch91], the pairing-based Boneh-Boyen-Shacham (BBS+) [BBS04], the RSA-based Camenisch-Lysyanskaya (CL) [CL03], the class of partitioned signatures [BSW06], and the CDH-based strongly unforgeable Waters signature scheme (Waters+) [BSW06].

Adaptor Signatures for Dichotomic Signatures. We then explore how the abstraction of dichotomic signatures contributes to the development of an adaptor signature for the hard relation $\mathsf{Rel} = \{(Y, y) : Y = \mathsf{OWF}(y)\}$. For simplicity, we omit the values sk and m from the function descriptions and describe only the random values used.

- To pre-sign, a message m w.r.t. a statement Y, and a signing key sk, we compute Σ_2 using randomness r but compute Σ_1 using a combined randomness $\mathsf{OWF}(r) \cdot Y$. That is, a pre-signature $\widetilde{\sigma} = (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ has the form

$$\widetilde{\sigma}_1 = \Sigma_1(\mathsf{OWF}(r) \cdot Y);\ \widetilde{\sigma}_2 = \Sigma_2(r).$$

- To adapt a pre-signature $\widetilde{\sigma} := (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ with witness y, we compute $\sigma := (\sigma_1, \sigma_2)$, where $\sigma_1 = \widetilde{\sigma}_1$, and $\sigma_2 = \widetilde{\sigma}_2 + y$. Note that by the homomorphism property of Σ_2, σ is a valid signature on m under the randomness $r + y$.
- To extract from a pre-signature $\widetilde{\sigma} = (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ and a corresponding adapted signature $\sigma = (\sigma_1, \sigma_2)$, we compute $\sigma_2 - \widetilde{\sigma}_2 = y$. The correctness of y follows since $\widetilde{\sigma}$ was adapted using a valid witness y by computing $\sigma_2 = \widetilde{\sigma}_2 + y$.
- For convenience's sake, we omit the pre-verification process and direct the reader to Construction 1 for a more elaborate discussion.

We refer to these adaptor signatures as *dichotomic adaptor* signatures. We now proceed to discuss the security proof.

Proving Security. Ideally, to prove the unforgeability security of adaptor signatures, we would like to reduce it to the unforgeability of the underlying dichotomic signature scheme. Any such reduction would need to simulate the signing as well as the pre-signing oracle for an adaptor signature adversary, *while only having access to the signing oracle in the unforgeability game of the dichotomic signature scheme.* However, it is not clear whether such a task is feasible. To elaborate, consider the following natural strategy to simulate the pre-signing oracle that takes as input a private key sk and an adversarially chosen message m and statement Y. To compute a pre-signature on this pair, the reduction may query its signing oracle to learn a signature σ on m. Note that this signature σ is a tuple (σ_1, σ_2) where $\sigma_1 = \Sigma_1(\mathsf{OWF}(r))$, and $\sigma_2 = \Sigma_2(r)$ (we again omit the values sk, and m). For this tuple, r is the random coins chosen by the signing oracle, which is kept secret from the reduction. If the reduction had access to the witness

y for Y, then it can easily turn σ into a valid pre-signature by computing $\widetilde{\sigma} := (\sigma_1, \sigma_2 - y)$. However, since Y is adversarially chosen and from a hard relation, it seems unlikely that the reduction knows the witness. In other words, it seems that the reduction must break the hard relation in order to simulate the pre-sign oracle.

To overcome this challenge, the most common approach adopted in known security proofs for adaptor signature schemes is to rely on the programmability of the random oracle, which we want to avoid. Another approach is to modify the hard relation to now contain the tuple (Y, π) as the statement where π is a NIZK proof-of-knowledge that certifies knowledge of the witness y. Then, the reduction can extract the witness y using the proof-of-knowledge extractor. However, NIZK proof-of-knowledge themselves either require a random oracle model [Fis05] or a trusted-setup [BFM88], which should be avoided in a decentralized setting like ours.

Transparent Reductions. Our key idea to overcome this challenge without relying on idealized models is to open up the reduction that bases the unforgeability of the underlying dichotomic signature scheme to some (possibly interactive) hardness assumption; henceforth called the *unforgeability reduction*. We observe that if the unforgeability reduction satisfies certain structural properties, then we can use its code in a non-black-box way to devise a reduction that bases the security of the adaptor signatures directly on the hardness assumption. Towards this, we introduce the notion of *transparent reductions*[1] which essentially capture the properties required from the unforgeability reductions.

More specifically, transparent reductions disclose three different algorithms as shown in Fig. 2. A *simulated key-generation* algorithm, a *simulated signing*, and a *break* algorithm that performs the functions as the names suggest: simulate key generation, simulate the signing oracle, and compute the solution for the hard problem, respectively. The adversary $\mathcal{A}_{\mathsf{Sign}}$ that breaks the unforgeability of the signature scheme receives as input the public output of the simulated key-generation algorithm and has access to the simulated signing algorithm as an oracle. When $\mathcal{A}_{\mathsf{Sign}}$ eventually outputs a valid forgery for the unforgeability game, the break algorithm is used to convert this forgery into a solution for the instance of the hard problem.

For the first step of our proof strategy, we reuse the code of these interfaces in a non-black-box way to provide a signing, key-generation, and break algorithm to the adversary $\mathcal{A}_{\mathsf{AS}}$ of the security of the adaptor signature scheme. In the second step, we have to simulate pre-signatures for $\mathcal{A}_{\mathsf{AS}}$, which we do using a property of transparent reductions we call *simulatability*. Simulatability guarantees that having as input the simulated signing key simsk, a message m, and a statement Y, the transparent reduction can compute a simulated pre-signature $\widetilde{\sigma}$ that looks indistinguishable from an honestly computed pre-signature. Since computing a signature and massaging it to a pre-signature does not seem feasible as outlined above, the reduction can use the power of the simulated signing key to *directly* compute a simulated pre-signature w.r.t. m and Y. With this direct computation, the reduction does not need to break either the hardness of the relation or the unforgeability of the signature scheme. Using the power of simulatable transparent reductions, a challenger can provide all needed oracles to an adversary *without* relying on the random oracle model. We show in Section 8, that these simulated oracles lead to

[1]We call these reductions transparent since we can "see-through" the reduction and use its code in a non-black-box way.

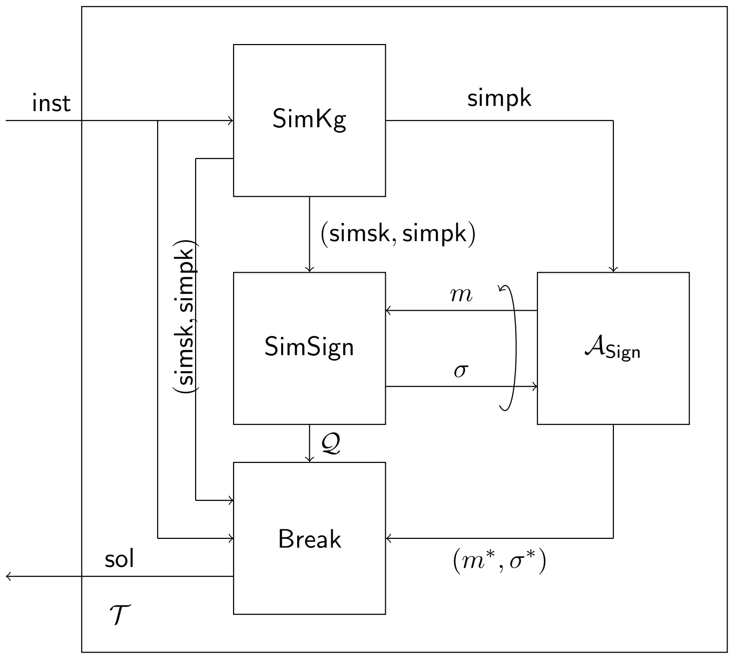

Figure 2: Visualization of a transparent reduction that reduces unforgeability to a non-interactive hard problem. The instance is a random instance of the non-interactive hard problem.

a security reduction that breaks the hardness of an underlying hard problem whenever an adversary wins the extractability experiment. This security reduction allows us to obtain the first adaptor signatures in the standard model, which are based on dichotomic adaptor signatures in the standard model. Besides from finding adaptor signatures in the standard model, transparent reductions allow us to prove the security for adaptor signature schemes based on a variety of assumptions using a single construction: We find dichotomic adaptor signatures based on strong RSA, pairings, CDH, and many more (c.f. Section 8).

We believe that transparent reductions are meaningful apart from proving the security of dichotomic adaptor signature schemes due to the following two reasons. First, whenever a primitive adds functionality to a signature scheme (e.g., verifiable encryption of signatures (VES)), it is not a priori clear whether the security of the new primitive can be reduced to the unforgeability of the signature scheme. In such a case, transparent reductions can bootstrap the proving process by removing redundancy: Instead of reducing the hardness of the primitive to the underlying problem, which requires simulating keys, a signing oracle, and providing a break algorithm, using a transparent reduction leverages the algorithms of the already existing reduction from the unforgeability of the signature scheme to the hard problem. Second, transparent reductions for signature schemes open a much broader field for transparent reductions for arbitrary cryptographic primitives, like encryption schemes or message authentication codes. Considering these possible applications, we believe that transparent reductions have future use for any scenario in which a non-black-box reduction removes redundancy from proofs.

Although our proof technique that relies on the new notion of transparent reductions is undoubtedly complex, it streamlines the verification process for subsequent constructions. That is, we need to check if only a few properties hold for the signature schemes' security reduction instead of reducing the security of the final adaptor construction to the underlying hard problem from scratch. Our work has done the bulk of the proof work. What is left to find secure adaptor signatures for a given signature scheme Σ is checking if the signature scheme is dichotomic and if there exists a simulatable transparent reduction from the unforgeability of Σ to the hardness of an underlying hard problem. In practice, many algebraic signature schemes are dichotomic and have a simulatable transparent reduction, as we show in Section 8.

2.4 New Instantiations of Secure Adaptor Signatures

All currently known adaptor signatures rely on the power of the random oracle model to achieve security. Furthermore, the only known security proof holds w.r.t. the security model provided by Aumayr *et al.* [Aum+21], that turns out to be insufficient for the novel applications as we show in Section 4. And beyond ID-based signature schemes, it is not clear whether we can generically find adaptor signatures for a given signature scheme. We address these issues by providing a general framework for building adaptor signatures that leads to secure instances w.r.t. our definitions in the standard model.

Adaptor signatures in the standard model are desirable since the goal of modern cryptography is to use as few and as weak assumptions as possible. Furthermore, many standardized signatures that are used in a wide variety of applications are in the standard model. A famous example is the Boneh-Boyen-Shacham (BBS+) signature scheme [BBS04]. The BBS+ signature scheme is the basis for anonymous credentials (AC) and several other privacy-preserving protocols [BL09; CDL16a; ASM06; Tsa+07], and is also standardized and deployed in many real-world systems [Loo+23; Hyp; Fin]. Our work shows how to build adaptor signatures for BBS+, thereby connecting the fairness properties of adaptor signatures with the privacy-preserving properties of BBS+.

Besides BBS+, we also show how to find adaptor signatures for many existent strongly unforgeable signature schemes Σ with two easy checks: First, one has to check if Σ is dichotomic, and second, if the security reduction $\mathcal{R}$ from the SUF-CMA security of Σ to an underlying hard problem is transparent and simulatable. We identify that the signature schemes Camenisch-Lysyanskaya (CL) [CL03], the class of partitioned signatures [BSW06], and the strongly unforgeable Waters signature (Waters+) [BSW06] are all dichotomic and have simulatable transparent reductions. Hence, we can build adaptor signatures for these signature schemes and find the first three secure adaptor signatures in the standard model. The security of these adaptor signatures is based on well-known (mild) assumptions, such as the CDH assumption or the strong RSA assumption, rather than the programmability of a heuristic model.

Furthermore, we show that all existent ID-based adaptor signature schemes also align with our framework (c.f. Section 8.5). Since our framework works generically, we do not have to prove the security of each of our new instantiations separately. This means that the strong security definitions for adaptor signatures, as discussed in Section 5, carry over to any dichotomic signature scheme with a simulatable transparent reduction. Since ID-based signatures are dichotomic and have a simulatable transparent reduction, our framework also implies the security of Schnorr adaptor signatures w.r.t. to the stronger security definitions of Dai *et al.*. This renders the current implementations for protocols based on Schnorr adaptor signatures secure.

Organization. This paper is organized as follows: We defer the basic preliminaries to Section 3, which includes the security definitions of adaptor signatures of Aumyar *et al.* [Aum+21]. In Section 4, we describe the gaps we identify, including presenting counter-example schemes, and discuss the insecure instantiations of several applications. Then, in Section 5, we propose stronger security definitions for adaptor signatures. We define our abstraction of dichotomic signatures in Section 6 and formalize transparent reductions in Section 7. Then, we discuss our secure dichotomic adaptor signature constructions and show how to build adaptor signatures for BBS+ Waters+, CL+, and Schnorr signatures in Section 8.

3 Preliminaries

By $x \leftarrow\$ X$, we denote the uniform sampling of x from the set X, and λ represents the security parameter. By $y \leftarrow \mathsf{A}(x; r)$ we denote a probabilistic polynomial time (PPT) algorithm A that on input x and randomness r, outputs y. When A is a deterministic polynomial time (DPT) algorithm, we use the notation $x := \mathsf{A}(y)$. We call a function $\nu : \mathbb{N} \to \mathbb{R}$ negligible in n if for every $k \in \mathbb{N}$, there exists $n_0 \in \mathbb{N}$ s.t. for every $n \geq n_0$ it holds that $|\nu(n)| \leq n^{-k}$. We use the function **assert** to check if a condition holds. The function call **assert** $\mathsf{condition}$ aborts the game in which it is called, if $\mathsf{condition}$ evaluates to false.

3.1 Homomorphic Quasi-Injective One-Way Functions

A one-way function $\mathsf{OWF} : \mathcal{D}_\mathsf{D} \to \mathcal{D}_\mathsf{S}$ is a function that is computable in polynomial-time, but given a value $Y \in \mathcal{D}_\mathsf{S}$ that is generated via $y \leftarrow\$ \mathcal{D}_\mathsf{D}; Y := \mathsf{OWF}(y)$ it is computationally hard to find a pre-image y' such that $\mathsf{OWF}(y') = Y$.

Definition 1 (One-way function)**.** *The function* $\mathsf{OWF} : \mathcal{D}_\mathsf{D} \to \mathcal{D}_\mathsf{S}$ *is* one-way*, if:*

Computability $Y := \mathsf{OWF}(y)$**.** *There exists an efficient algorithm that takes as an element $y \in \mathcal{D}_\mathsf{D}$ and computes $\mathsf{OWF}(y) = Y$.*

One-wayness. *For any* PPT *adversary $\mathcal{A}$, any randomly chosen value $y \leftarrow \mathcal{D}_\mathsf{D}$ with $\mathsf{OWF}(y) = Y$, the probability*

$$\Pr[\mathsf{OWF}(y') = Y | y' \leftarrow \mathcal{A}(Y)]$$

is negligible in λ.

We call OWF *homomorphic* if it carries over the algebraic structure from the domain to the support.

Definition 2 (Homomorphic Function)**.** *Let $\mathcal{D}_\mathsf{D}$ be an additive group and $\mathcal{D}_\mathsf{S}$ be a multiplicative group. A function $\mathsf{OWF} : \mathcal{D}_\mathsf{D} \to \mathcal{D}_\mathsf{S}$ is* homomorphic*, if for all $a, b \in \mathcal{D}$, it holds that $\mathsf{OWF}(a + b) = \mathsf{OWF}(a) \cdot \mathsf{OWF}(b)$.*

We choose additive and multiplicative groups for better readability. The definition naturally carries over to groups with arbitrary operators.

If we want the pre-image of a value $Y = \mathsf{OWF}(y)$ to be unique (with overwhelming probability), we assume OWF to be *quasi-injective*.

Definition 3 (Quasi-Injective Function). *A function* $\mathsf{OWF} : \mathcal{D}_{\mathsf{D}} \to \mathcal{D}_{\mathsf{S}}$ *is* quasi-injective*, if from* $\mathsf{OWF}(a) = \mathsf{OWF}(b)$ *it follows, that* $a = b$ *with overwhelming probability.*

3.2 Hard Relations

A relation Rel is a mapping defined as $\mathsf{Rel} : \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}} \to \{0, 1\}$ where $\mathcal{D}_{\mathsf{S}}$ is the space of statements and $\mathcal{D}_{\mathsf{W}}$ is the space of witnesses. Let $Y \in \mathcal{D}_{\mathsf{S}}$ be a statement and $y \in \mathcal{D}_{\mathsf{W}}$ be a witness. Rel maps (Y, y) to 1 if and only if y is a witness for the statement Y. The relation is hard if, with only the statement Y given, it is computationally infeasible to compute a witness y such that the relation is satisfied. For practical reasons, verifying the validity of a witness/statement pair and sampling instances (Y, y) of the relation should be computationally easy. We recall the notion of hard relations in **??**. We often build hard relations from one-way functions that we formalize as follows:

Definition 4 (Canonical Hard Relation). *Let* $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ *be a one-way function and* $\mathsf{Rel} : \mathcal{D}_{\mathsf{R}'} \times \mathcal{D}_{\mathsf{R}} \to \{0, 1\}$ *be a hard relation that is defined via* $\mathsf{Rel}(Y, y) = 1$ *if and only if* $Y = \mathsf{OWF}(y)$*. We call this relation a* canonical hard relation *for the function* OWF *and denote it via* $\mathsf{Rel}_{\mathsf{OWF}}$*.*

Hard Relations With Auxiliary Information. We generalize the notion of hard relations to hard relations with auxiliary information. Our definition of a hard relation with auxiliary information, denoted by $\mathsf{Rel}_{\mathsf{aux}}$, is based on an underlying hard relation Rel' and an underlying signature scheme Σ. Given a statement-witness pair $(Y, y) \in \mathsf{Rel}'$, the auxiliary information aux can be computed using the algorithm AuxGen on input the witness y. We define private and public decidability to determine if a pair (Y, aux) is well formed. Even with this auxiliary information and the signing key sk, for a statement $Y_{\mathsf{aux}} := (Y, \mathsf{aux})$ of the relation with auxiliary information $\mathsf{Rel}_{\mathsf{aux}}$, it remains hard to compute a witness y, such that $\mathsf{Rel}'(Y, y) = 1$.

Definition 5. *Let* $\mathsf{Rel}' \subseteq \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ *be a hard relation with statement/witness pairs* $(Y, y) \in \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{W}}$ *and* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a signature scheme. Let* $\mathsf{Rel}_{\mathsf{aux}} : \mathcal{D}_{\mathsf{aux}} \times \mathcal{D}_{\mathsf{S}} \to \{0, 1\}$ *be an efficiently computable relation. We say that the relation* $\mathsf{Rel} \subseteq \mathcal{D}_{\mathsf{S}} \times \mathcal{D}_{\mathsf{aux}} \times \mathcal{D}_{\mathsf{W}}$ *is a hard relation with auxiliary information w.r.t.* Σ *if:*

1. *Auxiliary Information Computation. There exists a* auxiliary information computation *algorithm* $\mathsf{AuxGen}(y, Y)$ *that on input a witness* $y \in \mathcal{D}_{\mathsf{W}}$ *for a statement* Y *outputs an auxiliary information* aux*, such that* $(Y, \mathsf{aux}) \in \mathsf{Rel}_{\mathsf{aux}}$*.*

2. *Hardness. For all* PPT *adversaries* $\mathcal{A}$*, there exists a negligible function* ν*, such that* $Pr[\mathsf{Rel}(Y, \mathsf{aux}, y^*) = 1 | (Y, y) \leftarrow \mathsf{RGen}(\lambda); \mathsf{aux} := \mathsf{AuxGen}(y); y^* \leftarrow \mathcal{A}(Y, \mathsf{aux}, \mathsf{sk})] \leq \nu(1^\lambda)$*, where the probability is derived by the random choice of* RGen *and* $\mathcal{A}$*.*

3. *(Public/Private) Decidability. There exists an efficient decide algorithm* $\mathsf{AuxVrfy}$ *that on the input of a (signing key* sk *of* Σ*) statement* Y*, and auxiliary information* aux *checks if* $(Y, \mathsf{aux}) \in \mathsf{Rel}_{\mathsf{aux}}$*. If* sk *is required, we call it private decidability; otherwise, public decidability.*

Note that every hard relation is a hard relation with empty auxiliary information. I.e., if $\mathcal{D}_{\mathsf{aux}} = \emptyset$, we can define $\mathsf{Rel}_{\mathsf{aux}}(\perp, \cdot) = 1$ and thus $\mathsf{Rel}(Y, \perp, y) = \mathsf{Rel}'(Y, y)$ is a hard relation. We can generally encode the auxiliary information into the statement by defining a new statement $Y' := (Y||\mathsf{aux})$. To simplify the notation, we assume that aux can efficiently be extracted from Y', and we will stick to the variable Y.

3.3 Adaptor Signatures

The formal definition of digital signatures can be found in Appendix A.1. In this section, we establish the definition of adaptor signatures in accordance with the security model presented in [Erw+21; Aum+21]. However, we highlight the inadequacy of their security model in Section 4 and direct the reader to Section 5 for more enhanced security definitions.

3.3.1 Definition of Adaptor Signatures

Adaptor signature schemes are defined for hard relations Rel and signature schemes Σ. They have a pre-sign algorithm that allows the signer to bind a statement Y of the hard relation Rel and a chosen message msg to some signature $\widetilde{\sigma}$. The pre-signature is publicly verifiable, proving that anybody possessing a corresponding witness y can adapt this pre-signature with the Adapt algorithm to a "full" signature σ. Furthermore, knowing a full signature σ and the corresponding pre-signature $\widetilde{\sigma}$, it is efficiently possible to extract the witness y for the encoded relation Rel using the $\mathsf{Extract}$ algorithm.

Definition 6 (Adaptor signature)**.** *An* adaptor signature scheme $\mathsf{AS}_{\Sigma,\mathsf{Rel}}$ *w.r.t. a signature scheme* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *and a hard relation* Rel *consists of a tuple of four algorithms* $\mathsf{AS}_{\Sigma,\mathsf{Rel}} = (\mathsf{pSign}, \mathsf{Adapt}, \mathsf{pVrfy}, \mathsf{Extract})$ *defined as:*

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

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

- $\underline{\sigma =: \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)}$. *The* adapting *algorithm is a* PPT *algorithm that on input a pre-signature* $\widetilde{\sigma}$ *and witness* y *for the statement* $Y \in L_{\mathsf{Rel}}$ *outputs an adapted signature* σ*.*

- $\underline{y =: \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)}$. *The* extracting *algorithm is a* DPT *algorithm that on input a pre-signature* $\widetilde{\sigma}$*, signature* σ *and statement* $Y \in L_{\mathsf{Rel}}$*, outputs a witness* y *such that* $(Y, y) \in \mathsf{Rel}$*, or* $\perp$*.*

We often omit the subscript from $\mathsf{AS}_{\Sigma,\mathsf{Rel}}$ writing AS to improve readability if the signature scheme Σ and the hard relation Rel are clear from the context. The correctness definitions look as follows:

Definition 7 (Pre-signature correctness). *An adaptor signature* AS *satisfies* pre-signature correctness*, if for all* $\lambda \in \mathbb{N}$ *and* $m \in \{0,1\}^{l_m}$*:*

$$\Pr\left[\begin{array}{c|l} \mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}) = 1 \wedge & (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda), \\ \mathsf{Vrfy}(\mathsf{vk}, m, \sigma) = 1 \wedge & (Y, y) \leftarrow \mathsf{RGen}(1^\lambda), \\ (Y, y') \in \mathsf{Rel} & \widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y) \\ & \sigma := \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y), \\ & y' := \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y) \end{array}\right] = 1.$$

Definition 8 (Pre-signature adaptability). *An adaptor signature scheme* AS *satisfies* pre-signature adaptability*, if for all* $\lambda \in \mathbb{N}$*, messages* $m \in \{0,1\}^*$*, statement/witness pairs* $(Y, y) \in \mathsf{Rel}$*, public keys* vk *and pre-signatures* $\widetilde{\sigma} \in \{0,1\}^*$ *we have* $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}) = 1$*, then* $\mathsf{Vrfy}(\mathsf{vk}, m, \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)) = 1$.

We stress that pre-signature adaptability is only defined for statements in the relation's language.

3.3.2 Current Insufficient Security Notions

Security of adaptor signatures is defined as unforgeability and witness extractability, which we recall in this subsection following [Erw+21; Aum+21].

Unforgeability. The unforgeability notion for adaptor signatures protects the signer because the adversary should not be able to forge a signature. Similar to the regular unforgeability notion, the adversary gets access to a signing oracle. In addition, it gets access to a pre-signing oracle, and it may query both oracles on messages of its choice. Unforgeability then demands that creating a forgery is hard for any efficient adversary even if it learned a *single* pre-signature on a possibly maliciously chosen message m^* for a given random statement $Y \in \mathcal{L}_{\mathsf{Rel}}$.

Definition 9 ($\mathsf{aEUF\text{-}CMA}$ security). *An adaptor signature scheme* AS *is* unforgeable *(or* $\mathsf{aEUF\text{-}CMA}$ *secure) if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* ν *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}}(\lambda) = 1\right] \leq \nu(\lambda),$$

where the definition of the experiment $\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}}$ *is given in Fig. 3 and the probability is taken over the random choices of all probabilistic algorithms.*

In this definition, the adversary can only learn *a single pre-signature on the challenge message.* Additionally, the adversary can only access witnesses of statements generated by himself. These restrictions do not accurately replicate an adversary's perspective in typical adaptor signature applications.

```
$\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}}(\lambda)$
1 : $\mathcal{Q} := \emptyset, (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $m^* \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
3 : $(Y, y) \leftarrow \mathsf{RGen}(1^\lambda)$
4 : $\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m^*, Y)$
5 : $\sigma^* \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\widetilde{\sigma}, Y)$
6 : return $(m^* \notin \mathcal{Q} \wedge \mathsf{Vrfy}(\mathsf{vk}, m^*, \sigma^*))$

$\mathsf{Sign}(\mathsf{sk}, m)$
1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
2 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
3 : return $\sigma$

$\mathsf{pSign}(\mathsf{sk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
2 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
3 : return $\widetilde{\sigma}$
```

Figure 3: The Game $\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}}$.

Witness Extractability. Informally, witness extractability protects the signer and guarantees that a malicious verifier can not use a pre-signature $\widetilde{\sigma}$ w.r.t. a statement Y to produce a valid signature σ without revealing a witness y for Y. So, an adversary wins the security game of witness extractability if he can provide a full signature that verifies but was never queried on the signing oracle and does not allow for extraction with a pre-signature on the same message from the pre-sign oracle.

Definition 10 (Witness extractability). *An adaptor signature scheme* AS *is* witness extractable *if for every* PPT *adversary $\mathcal{A}$, there exists a negligible function ν such that for every $\lambda \in \mathbb{N}$*

$$\Pr[\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}}(\lambda) = 1] \leq \nu(\lambda),$$

where the experiment $\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}}(\lambda)$ is defined in Fig. 4 and the probability is taken over the random choices of all probabilistic algorithms.

```
$\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}}(\lambda)$
1 : $\mathcal{Q} := \emptyset, (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $(m^*, Y^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
3 : $\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m^*, Y^*)$
4 : $\sigma^* \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\widetilde{\sigma})$
5 : $y^* := \mathsf{Extract}(\mathsf{vk}, \sigma^*, \widetilde{\sigma}, Y^*)$
6 : return $(m^* \notin \mathcal{Q} \wedge (Y^*, y^*) \notin \mathsf{Rel} \wedge \mathsf{Vrfy}(\mathsf{vk}, m^*, \sigma^*))$

$\mathsf{Sign}(\mathsf{sk}, m)$
1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
2 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
3 : return $\sigma$

$\mathsf{pSign}(\mathsf{sk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
2 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
3 : return $\widetilde{\sigma}$
```

Figure 4: The game $\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}}(\lambda)$.

The main difference between unforgeability and witness extractability is that in the game of witness extractability, the adversary provides the statement Y to the challenger.

Yet, this capability does not expose a trivial winning condition if the adversary adapts the pre-signature using a witness it already knows since $\mathcal{A}$ only wins the witness extractability game if the provided forgery does not reveal a witness for Y.

We observe two problems with the current witness extractability security definition: Witness extractability allows adaptor signature schemes where *two full signatures* can be obtained by adapting *one single pre-signature*, as long as they both extract to a valid witness. In addition, witness extractability only covers *a single execution of an adaptor signature protocol*, as the adversary never learns an adapted pre-signature on a statement without learning the witness before.

These security definitions lead to the following definition of a secure adaptor signature scheme:

Definition 11 (Secure adaptor signature scheme). *An adaptor signature scheme* AS *is secure if it is unforgeable and witness extractable.*

4 Security Gaps in Adaptor Signature Applications

This section shows three shortcomings in the current formalization of adaptor signatures. We provide simple counterexamples of adaptor signatures that are secure in the definitions of [Erw+21; Aum+21] but result in insecure real-world applications. To address each shortcoming, we follow a three-step approach. We first explain the application it affects and its key building blocks. Next, we show how these building blocks are instantiated using adaptor signatures. In the final step, we construct an adaptor signature scheme that trivially breaks the application while being secure w.r.t. the definitions of [Erw+21; Aum+21].

We wish to emphasize that the security gaps we identify do not affect the security of the concrete constructions used in these applications. Instead, we aim to highlight gaps in the definitions of adaptor signatures that potentially lead to *bad* instantiations, affecting the application's overall security.

4.1 Breaking Coin-Mixing Using Malleable Pre-Signatures

Glaeser et al. presented a protocol for coin-mixing at CCS'22 that utilizes blind conditional signatures (BCS) as a foundational component [Gla+22]. The authors demonstrate the instantiation of BCS with adaptor signatures. However, we show through a counter-example that a black-box construction of BCS from any adaptor signature scheme is impossible in general. For this, we give our counter-example that satisfies current adaptor signature definitions but yields an insecure BCS scheme, thus compromising the security of the coin-mixing protocol $\mathsf{A^2L^+}$ as formulated in [Gla+22]. In fact, the cryptographic primitives used in the recent work of Qin et al. [Qin+23] also suffers from the same drawback. But for simplicity, we will be using [Gla+22] to demonstrate the security gap. The crux of our attack is the exploitation of pre-signature malleability. A malleable pre-signature $\widetilde{\sigma}$ on a public key vk, message m, and statement Y can be converted into a distinct pre-signature $\widetilde{\sigma}'$ that verifies on the same public key, message, and statement, but adapts to a signature different from the one that we would get from adapting $\widetilde{\sigma}$.

In the following, we review the essential components of blind conditional signatures, along with the unforgeability definition for BCS. Additionally, we present a simple adaptor signature scheme that results in an insecure (i.e., forgeable) BCS scheme when instantiated with the construction of Glaeser et al. [Gla+22].

Blind Conditional Signatures (BCS). A blind conditional signature scheme is defined w.r.t. a signature scheme $\Sigma := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ and consists of the algorithms and protocols $(\mathsf{Setup}, \mathsf{PPromise}, \mathsf{PSolver}, \mathsf{Open})$. To understand the attack, it is sufficient to focus on the protocols $\mathsf{PPromise}$ and $\mathsf{PSolver}$ that are executed within the unforgability experiment of BCS. We refer to [Gla+22] for the complete definitions:

- The puzzle promise protocol $\mathsf{PPromise}$ is an interactive protocol between two users named H (the Hub) and Bob (user Bob). Both parties receive keys and a message m as input, and at the protocol's end, Bob outputs a puzzle τ and H gets nothing:

$$(\perp, \tau) \leftarrow \mathsf{PPromise}\,\langle \mathsf{H}(\star, m), \mathsf{Bob}(\star, m)\rangle\,.$$

  Intuitively, the puzzle locks the signature σ on the message m.

- The interactive puzzle-solving protocol, running between users H and Bob, is a protocol in which Bob learns a signature on a message that is "locked" in a puzzle τ; the user H also learns the signature (along with some other secret s that is irrelevant to our attack and will be omitted):

$$((\sigma, \star), \sigma) \leftarrow \mathsf{PSolver}\,\langle \mathsf{H}(\star, m), \mathsf{Bob}(\star, m, \tau)\rangle\,.$$

In the following, we focus on the one-time unforgeability game, which gives the attacker only one-time access to each oracle. Intuitively, (one-time) unforgeability requires that a malicious Bob cannot forge signatures after learning *a single* signature. Here, we focus on strong unforgeability, where the message may be the same, but the signatures must be different, i.e., the adversary wins if he makes a single query to the oracle and manages to compute a different signature (on the same message). This game is simpler and sufficient for our attack. Furthermore, it makes our result stronger because not even one-time unforgeability can be achieved in general. To formalize this notion, the adversary $\mathcal{A}$ (playing the role of Bob) is given access to a $\mathsf{PPromise}$ and a $\mathsf{PSolver}$ oracle. Let us denote by $(\mathsf{vk}, m, \sigma_0), (\mathsf{vk}, m, \sigma_1)$ the final output of $\mathcal{A}$. Intuitively, the attacker $\mathcal{A}$ breaks the unforgeability if it satisfies the following conditions (other conditions specified in (c.f. [Gla+22], Figure 7) are omitted as they are irrelevant to our attack):

1. $\mathcal{A}$ submitted m to the promise oracle $\mathsf{PPromise}$ and received vk from $\mathsf{PPromise}$; i.e., $(\mathsf{vk}, m) \in \mathcal{L}$;

2. all signatures output by $\mathcal{A}$ must be valid, i.e., $\mathsf{Vf}(\mathsf{vk}, m, \sigma_0) = \mathsf{Vf}(\mathsf{vk}, m, \sigma_1) = 1$;

3. the tuples must be distinct, i.e.,$(\mathsf{vk}, m, \sigma_0) \neq (\mathsf{vk}, m, \sigma_1)$;

4. and $\mathcal{A}$ queried the solving oracle only once, i.e., $Q \leq q - 1$, where Q is the number of oracle queries and $q - 1$ the number of forgeries returned. In our one-time case, we have $q = 2$.

BCS **Instantiation Using Adaptor Signatures.** Glaeser et al. instantiate BCS using an adaptor signature scheme AS. We concentrate on the construction of the promise and solve protocol applicable to our attack and refer to [Gla+22] for formal construction.

- At the end of the $\mathsf{PPromise}$ protocol, Bob returns a puzzle τ that contains a message m, a statement Y, and a pre-signature $\widetilde{\sigma}$ (and some other elements irrelevant for our attack).
- In the solve protocol $\mathsf{PSolver}$, the party H receive the pre-signatures, adapts it to a full signature σ, and returns σ.

Glaeser et al. provide the following proposition, which states the security of the $\mathsf{A^2L^+}$ protocol, assuming the security of the underlying adaptor signature scheme.

Proposition 1 (Theorem 4.9 from [Gla+22])**.** *Let Π_E be a linear-only encryption scheme, Σ be a strongly unforgeable signature scheme,* Rel *be a hard relation, and let Π_{NIZK} be a sound NIZK proof system. Let* AS *be a secure adaptor signature scheme for Σ and* Rel*. Assuming the hardness of* one more discrete-log *(OMDL) problem, the* $\mathsf{A^2L^+}$ *instantiation from [Gla+22] is a secure blind conditional signature scheme.*

Breaking Blind Conditional Signatures from Adaptor Signatures. Contrary to this proposition, we claim in Theorem 1 that Proposition 1 does not hold in general, and we prove this claim in two steps: First, we create an adaptor signature scheme AS' that adheres to current definitions but enables an adversary to obtain two distinct valid signatures from a single pre-signature via malicious adapting. Second, we illustrate how an attacker can break the unforgeability of the BCS construction of [Gla+22] if it is instantiated using AS'. Our proof requires a natural property of the main building block of AS', which we call *unconnected* adaptor signatures defined below. To the best of our knowledge, all currently known adaptor signature schemes (e.g., Schnorr-based) are unconnected.

Theorem 1. *There exists a secure adaptor signature scheme* AS'*, such that the* BCS *instantiation from [Gla+22] does not satisfy (one-time) unforgeability.*

Unconnected Adaptor Signatures. Our counterexample adaptor signature scheme needs to be unconnected, which means that for all $\lambda \in \mathbb{N}$, all keys $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\lambda)$, all messages m, all statement-witness pairs $(Y, y) \leftarrow \mathsf{RGen}(\lambda)$, and all elements $r_1 \neq r_2 \leftarrow\$ \mathcal{D}_{\mathsf{R}}$, there exists a negligible function ν, such that the probability

$$\Pr[\mathsf{Adapt}(\mathsf{vk}, \mathsf{pSign}(\mathsf{sk}, m, Y; r_1), y) = \mathsf{Adapt}(\mathsf{vk}, \mathsf{pSign}(\mathsf{sk}, m, Y; r_2), y)] \leq \nu(\lambda).$$

Intuitively, this property ensures that two pre-signatures generated with independent randomness but on the same message and statement result in two distinct signatures with overwhelming probability.

Malleable Pre-Signatures. We now build a secure adaptor signature scheme AS' that has malleable pre-signatures and is based on an unconnected adaptor signature scheme AS. Each pre-signature of AS' is a pair of two distinct pre-signatures of AS on the same message and statement. The pre-verify algorithm of AS' runs $\mathsf{AS.pVrfy}$ on both pre-signatures components separately. The adapt algorithm of AS' adapts the first component of the pre-signature using $\mathsf{AS.Adapt}$. The extract algorithm tries to extract a valid witness for the statement Y using the signature and the first pre-signature component by calling $\mathsf{AS.Extract}$. If this fails, it tries the same using the signature and the second pre-signature component. A formal description of the transformation from AS to AS' is given in Fig. 5 and we provide a formal security analysis of AS' in Lemma 23.

```
pSign'(sk, m, Y)
  1 : $(r_1, r_2) \leftarrow\$ \mathbb{Z}_p^2$
  2 : $\widetilde{\sigma}_1 \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y; r_1)$
  3 : $\widetilde{\sigma}_2 \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y; r_2)$
  4 : return $(\widetilde{\sigma}_1, \widetilde{\sigma}_2)$

pVrfy'(vk, m, Y, σ̃)
  1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
  2 : $b_1 := \mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}_1)$
  3 : $b_2 := \mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}_2)$
  4 : return $b_1 \wedge b_2$

Adapt'(vk, σ̃, y)
  1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
  2 : $\sigma := \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}_1, y)$
  3 : return $\sigma$

Extract'(vk, σ̃, σ, Y)
  1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
  2 : $y =: \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}_1, \sigma, Y)$
  3 : if $(Y, y) \in \mathsf{Rel}$ return $y$
  4 : return $\mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}_2, \sigma, Y)$
```

Figure 5: An adaptor signature scheme AS' for which any valid pre-signature of AS' can be adapted to two different valid signatures.

Breaking the Unforgeability of BCS Using Malleable Pre-Signatures. The following adversary $\mathcal{A}$ breaks the (one-time) unforgeability of BCS if it is instantiated using AS'. $\mathcal{A}$ queries the $\mathsf{PPromise}$ oracle once on a random message m and learns a puzzle τ. The puzzle τ contains a pre-signature $\widetilde{\sigma}$ on m on a random statement Y. Furthermore, the adversary queries the PSolve oracle once on input the puzzle τ and learns a signature σ, which is the output of $\mathsf{Adapt}(\widetilde{\sigma}, y)$, where y is the corresponding witness of Y. With this pre-signature-signature pair, $\mathcal{A}$ can learn y using the $\mathsf{Extract}$ algorithm. Knowing the pre-signature $\widetilde{\sigma}$ and the corresponding witness y, the adversary parses $\widetilde{\sigma} := (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ and computes $\widetilde{\sigma}' := (\widetilde{\sigma}_2, \widetilde{\sigma}_1)$ by switching the order of the elements. Now, using the adapt algorithm Adapt' on input $(\widetilde{\sigma}', y)$, the adversary obtains a second distinct signature σ' on the message m. Note that with overwhelming probability, the second σ' is distinct from σ since AS is an unconnected adaptor signature scheme. This breaks the unforgeability of BCS.

4.2 Breaking Blind Hubs Using Unadaptable Adaptor Signatures

The second definitional gap stems from the fact that the security definitions do not hold for statementsthat are not in the language of the relation Rel, i.e., $Y \notin \mathcal{L}_{\mathsf{Rel}}$. This gap is not a problem for early applications, such as payment channels, because the user computing the statement also adapts the corresponding pre-signature. However, we show that this gap leads to devastating attacks in recent applications, such as coin-mixing, which allows the adversary to steal coins.

In the following, we first give a counter-example in which a valid pre-signature for a "non-language" statement cannot be adapted to a valid signature. Then, we show that this leads to an immediate loss of fairness in coin-mixing applications [Qin+23; Gla+22].

Flexible Blind Conditional Signatures. Qin *et al.* [Qin+23] propose the new notion of flexible blind conditional signatures (FBCS). FBCS are blind conditional signatures in which the hub H (cf. Section 4.1) only learns a commitment on a transaction rather than the transaction in plaintext. Similar to the security notions of ordinary BCS, a crucial security requirement for FBCS is *unlockability.* Unlockability guarantees that no malicious hub H can refrain coins by running a PPromise and a related PSolve. In the unlockability security experiment, an adversary $\mathcal{A}$ can run a single execution of PPromise and a related PSolve. $\mathcal{A}$ wins the experiment if he:

- Outputs a valid signature on a transaction message from Bob to $\mathcal{A}$;
- the puzzle open algorithm run by Bob does not reveal a valid signature from $\mathcal{A}$ to Bob.

FBCS Instantiation Using Adaptor Signatures. Qin *et al.* provide a construction for a FBCS based on the DLog relation and the ECDSA signature scheme, which also uses adaptor signature schemes for ECDSA. Due to the similarity of the FBCS construction and the BCS construction, we refer the reader to Section 4.1 for a discussion of this construction. The authors refer to this FBCS construction as BlindHub and formally claim its security:

Proposition 2 (Theorem 1 from [Qin+23], informal)**.** *Let Π_{Enc} be a linear-homormorphic encryption scheme, Π_{AS} a secure adaptor signature scheme, Π_{BAS} a secure BAS scheme, Π_{RSoRC} a secure signature on randomizable commitments scheme, Π_{NIZK} a sound proof system. If the one-more DLog assumption is assumed to be hard, then the* BlindHub *protocol is a secure, flexible blind conditional signature scheme.*

Breaking (Flexible) Blind Conditional Signatures from Adaptor Signatures. Contrary to this proposition, we claim that the BlindHub construction is insecure in general if it is instantiated using an unadaptable adaptor signature scheme.

Theorem 2. *If the* BlindHub *protocol of [Qin+23] is instantiated with a unadaptable adaptor signature scheme* AS' *and a relation that has malicious statements for valid witnesses, then it is in general not a secure flexible blind conditional signature scheme.*

```
$\mathsf{pSign}'(\mathsf{sk}, m, Y)$
1 : if $Y \notin \mathcal{L}_{\mathsf{Rel}}$ then
2 :     $\widetilde{\sigma} := \perp$
3 : else
4 :     $\widetilde{\sigma} := \mathsf{pSign}(\mathsf{sk}, m, Y)$
5 : return $\widetilde{\sigma}$

$\mathsf{pVrfy}'(\mathsf{vk}, m, Y, \widetilde{\sigma})$
1 : if $Y \notin \mathcal{L}_{\mathsf{Rel}}$ then
2 :     $ret = 1$
3 : else
4 :     $ret := \mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma})$
5 : return $ret$

$\mathsf{Adapt}'(\mathsf{vk}, \widetilde{\sigma}, y)$
1 : if $\widetilde{\sigma} = \perp$ then
2 :     $\sigma = \perp$
3 : else
4 :     $\sigma = \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)$
5 : return $\sigma$

$\mathsf{Extract}'(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)$
1 : if $\widetilde{\sigma} = \perp$ then
2 :     $y := \perp$
3 : else
4 :     $y := \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)$
5 : return $y$
```

Figure 6: An secure adaptor signature scheme AS', for which valid pre-signatures on malicious statements can not be adapted to a verifying signature.

We prove Theorem 2 in two steps: First, we show how to build unadaptable adaptor signatures. Second, we show how to break unlockability using unadaptable adaptor signatures.

Construction of an Unadaptable Adaptor Signature Scheme. Let AS be a secure adaptor signature for some signature scheme Σ and the relation Rel, such that the language $\mathcal{L}_{\mathsf{Rel}}$ is efficiently decidable: there exists an efficient algorithm that outputs 1 if $Y \in \mathcal{L}_{\mathsf{Rel}}$, and 0 otherwise. Given AS, we construct a new adaptor signature AS', that adheres to the same security properties of AS, and therefore also to the adaptability notion, but which trivially leads to an attack for statements not in the language of the relation Rel, i.e., $Y \notin \mathcal{L}_{\mathsf{Rel}}$. We model this malicious functionality by letting the pre-signing algorithm output $\perp$ for statements not in the language of the relation. The pre-verification algorithm accepts $\perp$ as a valid pre-signature if $Y \notin \mathcal{L}_{\mathsf{Rel}}$.

Therefore, $\perp$ is a valid pre-signature on a malicious statement, and thus Adapt can not adapt $\perp$ to a valid signature on the inputs of a public key vk and a valid witness y. A formal description of AS' is given in Fig. 6 and we prove the security of AS' in Lemma 24.

Breaking Unlockability. In a coin mixing setup, an adversary acting as hub H can break unlockability if the used adaptor signature scheme is unadaptable for dishonest statements: In the puzzle promise phase, the hub H samples a malicious statement Y' and computes a valid pre-signature on Y' by simply outputting $\perp$. The other party Bob accepts the puzzle since the pre-signature $\perp$ pre-verifies. Bob now runs the puzzle solve protocol together with H. When the puzzle is solved, H learns a signature and is paid by Bob. However, Bob can not adapt the pre-signature (i.e., $\perp$) provided by H even if he learns a valid witness for the statement Y', so Bob can not claim a payment by H.

The *concrete instantiations* provided in [Gla+22; Qin+23] avoid this kind of attack since the relation has only witnesses for statements that are well-formed, and H must prove the knowledge of a witness using a NIZK. Yet, Proposition 2 does not hold in general: If we instantiate the protocols with a relation, where maliciously chosen statements $Y \notin \mathcal{L}_{\mathsf{Rel}}$ can have valid witnesses, the hub H can prove knowledge of the witness while pre-signing w.r.t. a malicious statement. The language $\mathcal{L}_{\mathsf{Rel}} := \{((1, Y), y) | Y = g^y; y \in \mathbb{Z}_p\}$ yields such a hard relation. When instantiated with such a hard relation, the blind hub protocol is insecure since H can prove the knowledge of a witness for a maliciously computed statement of the form $(0, Y)$. This statement is clearly not in the language since the first bit is a 0.

4.3 Breaking VweTS Using Signature Leaky Pre-Signatures

Madathil et al. [Mad+22] introduced *verifiable witness encryption based on threshold signatures (VweTS)* at NDSS'23 that with applications in oracle-based conditional payments for blockchains. Their instantiation of VweTS uses adaptor signatures as one of the many building blocks. In this section, we present another gap in the definitions of adaptor signatures that leads to an attack against their instantiation of VweTS.In the following, we first recall relevant parts of their construction necessary to follow our attack. We then construct the adaptor signature scheme that satisfies the security definitions but leads to a devastating attack when used as a building block in the construction of VweTS.

Verifiable Witness Encryption Based on Threshold Signatures (VweTS). A VweTS scheme uses two signature schemes, namely, $\Sigma := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ and $\Sigma' := (\mathsf{KGen}', \mathsf{Sign}', \mathsf{Vf}')$ and provides the following efficient algorithms $(\mathsf{EncSig}, \mathsf{VfEnc}, \mathsf{DecSig})$. We refer to Σ' as the signature scheme and Σ as the transaction signature scheme. A VweTS protocol is run between an encryptor that encrypts transaction signatures on transaction messages into ciphertexts and a decryptor that decrypts these ciphertexts to obtain transaction signatures. The important part is that the decryption is not done using a decryption key but rather using a sufficient number of instance signatures on instance messages. An instance signature confirms the instance message by a third party identified by an instance verification key vk'. We call these third parties instance signers from now on.

To understand our attack, it is sufficient to focus on the protocols EncSig, and DecSig and we refer to [Mad+22] for a more formal description of VweTS.

- The signature encryption algorithm EncSig encrypts a tuple of signatures $(\sigma_j)_{j\in[M]}$ on transaction messages $(m_j)_{j\in[M]}$ to produce a ciphertext c. The ciphertext c is generated with respect to a tuple of instance verification keys $(\mathsf{vk}'_i)_{i\in[N]}$, and instance messages $(m'_j)_{j\in[M]}$ as well.
- The decryption algorithm DecSig takes as input a tuple of ρ (valid) instance signatures σ'_i (under the instance verification keys vk'_i) on an instance message m'_j. It returns the transaction signature σ_j on the transaction message m_j.

The main security property of a VweTS scheme is called one-wayness, which intuitively guarantees that no adversary can output a valid transaction signature forgery σ^* for a transaction message m_j encrypted in a VweTS ciphertext c without access to at least ρ number of valid instance signatures on the corresponding instance messages m'_j. To formalize one-wayness, an adversary $\mathcal{A}$ can corrupt up to $\rho - 1$ instance signers and has access to a signature encryption oracle that runs EncSig on adversarially chosen instance messages and transaction messages. We omit the other oracles, as our attack will not use them. Furthermore, our attack only queries the signature encryption oracle once. Eventually, $\mathcal{A}$ has to output a signature forgery σ^*, an index j^* that marks the j^*-th transaction message in its encryption oracle call. Since our attack only queries the signature encryption oracle once, $\mathcal{A}$'s output has to satisfy the following conditions to win the one-wayness experiment:

1. The adversary corrupted less than ρ instance signers;
2. the signature σ^* verifies under the challengers verification key vk and the $j^* - th$ transaction message m_{j^*}.

VweTS Instantiation Using Adaptor Signatures. Madathil *et al.* also presented VweTS constructions, which utilize adaptor signatures as a building block. Intuitively, the encryption of a transaction signature on a transaction message m_j consists of an adaptor pre-signature on m_j with respect to a random statement $Y_j := g^{y_j}$ for $y_j \leftarrow\$ \mathbb{Z}_p^*$. The witness y_j is verifiably secret-shared, and each share is encrypted using a special encryption scheme that is irrelevant to our context. The only information about this special encryption scheme we need to know here is that the j-th ciphertext can be decrypted given ρ instance signatures on the instance message m'_j. To decrypt a signature on the message m_j, one has to *adapt* the pre-signature on m_j to a valid signature using the witness y_j. Recall that the witness y_j is available by decrypting ρ number of the special ciphertexts given the instance signatures.

Madathil *et al.* proposed the security of the VweTS construction utilizing the security of the used adaptor signature scheme in a theorem, which we reiterate below as Proposition 3.

Proposition 3 (Theorem 1 from [Mad+22], informal)**.** *Let Σ and Σ' be secure signature schemes, WES be secure witness encryption based on signatures, AS be a secure adaptor signature scheme for Σ and the relation Rel. Let $\Pi := (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vrfy})$ be a NIZK proof system satisfying simulation soundness. Then, the VweTS construction from [Mad+22] achieves one-wayness.*

Breaking VweTS from Adaptor Signatures. Contrary to this proposition, we claim that there exists an adaptor signature scheme AS that is secure in accordance with Definition 11, but if the VweTS construction from [Mad+22] is instantiated with AS, one-wayness does not hold. We prove our claim in two steps. First, we construct a secure adaptor signature scheme AS, in which multiple pre-signatures on the same message can reveal information on an additional valid signature on the same message. Second, we show how an adversary wins the game $\mathsf{ExpOWay}$ with overwhelming probability if a VweTS is instantiated using AS'.

Theorem 3. *Let Σ, Σ', Π, and* WES *be as in Proposition 3. There exists a secure adaptor signature scheme* AS' *for the signature scheme* Σ *and the relation* Rel, *such that the* VweTS *construction from [Mad+22] instantiated with* AS' *does not achieve one-wayness.*

An Adaptor Signature with Leaky Pre-Signatures. In Definition 10 (witness extractability) and Definition 9 ($\mathsf{aEUF\text{-}CMA}$ security), the adversary $\mathcal{A}$ of the adaptor signature is allowed to query the pre-signing oracle on a challenge message m^* exactly one time. However, in the VweTS security game, the adversary $\mathcal{A}$ can query the EncSig oracle with a tuple of transaction messages $(m_0, \ldots, m_M)$. Thereby, $\mathcal{A}$ can learn at least two pre-signatures on the same message with different random adaptor statements for each.

Our counter-example is inspired by [DOY22]. Dai *et al.* propose a counter-example that shows a gap if an adversary can see two pre-signatures on the same message and the same statement. This counterexample can not break Proposition 3, since the VweTS construction samples a random statement for each pre-signature, and the probability of learning two pre-signatures on the same message is negligible. Furthermore, they use a pseudorandom function instead of a random function. As a key for the PRF, they also use the signing key sk. Since PRFs are not leakage-resilient in general, it can be detectable by an adversary if it learns $\widetilde{\sigma}, \mathsf{PRF}(\mathsf{sk}, m, Y)$, or $\widetilde{\sigma}, r$ for a random value $r \leftarrow\!\!\$ \{0,1\}^\ell$ (c.f. game hop between $\mathcal{G}_0$ and $\mathcal{G}_1$ in [DOY22], proof of their Theorem 3). Therefore, we do not assume their counterexample to be proven secure but follow the flavor of this counterexample in a provable manner.

To overcome the described issues, our counter-example uses hash functions modeled as random oracles instead of pseudorandom functions. Using the RO is reasonable since the VweTS constructions in [Mad+22] are based on Schnorr and ECDSA signatures that already utilize the random oracle model for security. On a high level, our counter-example adaptor signature scheme AS' pre-signs a message-statement pair (m, Y) using the underlying pre-sign algorithm of AS and by appending a random element r to this pre-signature. Depending on a coin toss, the random element is either $\mathcal{H}(\mathsf{sk}, m)$, i.e., the output of the random oracle on the input of the signing key sk and the message m or the XOR of $\mathcal{H}(\mathsf{sk}, m)$ and a valid signature on m. AS' is formally defined in Fig. 7.

This adaptor signature scheme AS' is secure w.r.t. Definition 11, as we validate in Lemma 25.

How an Adversary Can Break the One-Wayness of VweTS We now show how an adversary $\mathcal{A}$ can break the one-wayness of VweTS if the construction in [Mad+22] is instantiated using AS'. One way for an adversary $\mathcal{A}$ to win the game $\mathsf{ExpOWay}$ is to compute a verifying signature on a challenge message m_j by only querying the EncSig oracle and not querying the signature and the instance signature oracles related to m_j. Let us assume for simplicity that $M = 2$, where M is the size of the message tuples used in EncSig. Keeping this in mind, the adversary queries the oracle $\mathsf{EncSig}\mathcal{O}$ on random instance messages m'_0, m'_1, and on a tuple of transaction messages (m_0, m_0). Internally, the oracle computes two pre-signatures on the message m_0 with random statements and outputs the pre-signatures as part of the VweTS ciphertext c. Therefore, with probability $1/2$, the adversary can learn a valid signature on the message m_0 by XOR-ing the two random elements from the two pre-signatures returned by the oracle. In the generic case

```
pSign'(sk, m, Y)
  1 : $\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y)$
  2 : $b \leftarrow\!\$ \{0,1\}$
  3 : $r_0 \leftarrow \mathcal{H}(\mathsf{sk}, m)$
  4 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
  5 : $r_1 := r_0 \oplus \sigma$
  6 : return $(\widetilde{\sigma}, r_b)$

pVrfy'(vk, m, Y, σ̃)
  1 : $(\widetilde{\sigma}', \bot) := \widetilde{\sigma}$
  2 : return $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}')$

Extract'(vk, σ̃, σ, Y)
  1 : $(\widetilde{\sigma}', \bot) := \widetilde{\sigma}$
  2 : return $\mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}', \sigma, Y)$

Adapt'(vk, σ̃, y)
  1 : $(\widetilde{\sigma}', \bot) := \widetilde{\sigma}$
  2 : return $\mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}', y)$
```

Figure 7: A secure adaptor signature scheme AS', for which two pre-signatures on the same message can leak an additional signature.

of M being some arbitrary polynomial in the security parameter, the adversary can use the above strategy and win with overwhelming probability $1-(1/2)^{M-1}$. This breaks the one-wayness of adaptor signature-based VweTS constructions.

5 Enhanced Adaptor Signature Security Definitions

The attacks presented in Section 4 serve as clear indications that the security properties captured in the definition of adaptor signatures in [Erw+21; Aum+21] are insufficient for most applications. The work of Dai, Okamoto, and Yamamoto [DOY22] takes a significant step towards remedying this state of affairs by proposing stronger security properties that a good adaptor signature scheme should satisfy (over the properties captured by [Erw+21; Aum+21]).

In this section, we restate the properties of *extractability*, *unique extractability*, and *unlinkability* as defined in [DOY22]. In addition, we identify an additional security property for adaptor signatures called *pre-verify soundness*, which is important for certain applications such as (blind) Coin Mixing [Gla+22; Qin+23] and oracle-based payments [Mad+23]. All four properties are desirable, and a good adaptor signature scheme should satisfy all of them. However, as we will see later, some applications of interest may require only a subset of these four properties from the adaptor signature scheme.

5.1 Extractability

Extractability, proposed by Dai, Okamoto, and Yamamoto [DOY22], combines and extends the security properties of witness extractability (Definition 10) and adaptor unforgeability (Definition 9) to the so-called *multiple* query setting. In particular, the adversary is allowed to see multiple pre-signatures on the challenge message as well as pre-signatures on multiple honestly sampled statements. In contrast, in prior definitions, the adversary was restricted to seeing only a single pre-signature on the challenge message and an honestly sampled statement. In the multiple query setting, the adversary's task is to output a special forgery (m^*, σ^*) for which σ^* cannot be used to successfully extract a witness (breaking extractability). We formally state their definition below:

Definition 12 (Extractability)**.** *An adaptor signature scheme* AS *is extractable, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* ν *such that for every* $\lambda \in \mathbb{N}$

$$\Pr[\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}(\lambda) = 1] \leq \nu(\lambda) \ ,$$

where the experiment $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}$ *is described in Fig. 8, and the probability is taken over the random choices of all probabilistic algorithms.*

```
$\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}(\lambda)$
  $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda); b \leftarrow 1; \mathcal{S}, \mathcal{T} \leftarrow \emptyset$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{vk})^{\mathsf{NewY}(\lambda), \mathsf{Sign}(\mathsf{sk}, \cdot), \mathsf{pSign}(\mathsf{sk}, \cdot, \cdot)}$
  **assert** $\mathsf{Vrfy}(\mathsf{vk}, m^*, \sigma^*)$
  **assert** $(m^* \notin \mathcal{S})$
  **for** $(Y, \widetilde{\sigma}) \in \mathcal{T}[m^*]$
    **if** $(Y, \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma^*)) \in \mathsf{Rel}$ **then**
      $b \leftarrow 0$
  **return** $b$

$\mathsf{NewY}(\lambda)$
  $(Y, y) \leftarrow \mathsf{Rel.RGen}(1^\lambda)$; **return** $Y$

$\mathsf{Sign}(\mathsf{sk}, m)$
  $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{m\}$
  **return** $\sigma$

$\mathsf{pSign}(\mathsf{sk}, m, Y)$
  $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
  $\mathcal{T}[m] \leftarrow \mathcal{T}[m] \cup \{(Y, \widetilde{\sigma})\}$
  **return** $\widetilde{\sigma}$
```

Figure 8: The security game $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}(1^\lambda)$.

Extractability Prevents Leaky Pre-Signatures. Leaky pre-signatures, which we introduced in Section 4.3, allow an adversary to break the security of applications that use adaptor signatures in a multi-query setting: A single leaky pre-signature only reveals randomized information, whereas multiple pre-signatures on the same message reveal an additional signature on the same message. If we use an adversary that mimics the same strategy in the security game Ext of extractability, we can see that this adversary wins Ext with probability $1 - (1/2)^{|\mathcal{T}[m^*]-1|}$ rendering adaptor signatures with leaking pre-signatures insecure: The adversary queries the pre-sign oracle multiple times on the same message m^* and extracts a fresh signature on m^*. This fresh signature does not extract with any pre-signature and hence is successful in winning Ext. Therefore, the counter-example we provide in Section 4.3 cannot achieve extractability.

5.2 Unique Extractability

Unique extractability guarantees that any verifying pre-signature can be viewed as a commitment to *both* a *single* valid signature and a *single* witness. This means that no efficient adversary can compute a pre-signature $\widetilde{\sigma}$ on a message m and a statement Y, such that there exist two different signatures on m that both extract to a valid witness with $\widetilde{\sigma}$. More formally:

Definition 13 (Unique Extractability). *An adaptor signature scheme* AS *is unique extractable, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* ν *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\big[\mathsf{UniqueExtractability}_{\mathcal{A},\mathsf{AS}}(\lambda) = 1\big] \leq \nu(\lambda) \ ,$$

where experiment $\mathsf{UniqueExtractability}_{\mathcal{A},\mathsf{AS}}$ *is described in Fig. 9, and the probability is taken over the random choices of all probabilistic algorithms.*

```
UniqueExtractability_{A,AS}(λ)
1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $(m, Y, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
3 : assert $(\sigma \neq \sigma') \wedge \mathsf{Vrfy}(\mathsf{vk}, m, \sigma) \wedge \mathsf{Vrfy}(\mathsf{vk}, m, \sigma')$
4 : assert $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma})$
5 : $y \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma')$
6 : return $(Y, y) \in \mathsf{Rel} \wedge (Y, y') \in \mathsf{Rel}$

Sign(sk, m)
1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
2 : return $\sigma$

pSign(sk, m, Y)
1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
2 : return $\widetilde{\sigma}$
```

Figure 9: The security game $\mathsf{UniqueExtractability}_{\mathcal{A},\mathsf{AS}}(\lambda)$.

Unique Extractability Prevents Malleable Pre-Signatures. Unique extractability enforces a one-to-one relation between pre-signatures and adapted signatures. Therefore, we can see a pre-signature of an adaptor signature scheme that achieves unique extractability as a binding commitment to a single signature. The counter-example we present in Section 4.1 does not achieve unique extractability since each pre-signature can be adapted to two different full signatures that both extract. Such an adaptor signature scheme cannot achieve unique extractability.

(Strong) Full Extractability Dai *et al.* also propose two more security notions named (strong) full extractability. Full extractability facilitates the security proofs for adaptor signatures under the caveat of a more complex definition, and strong full extractability transfers the goals of strong unforgeability of signature schemes for adaptor signatures. Strong full extractability is implied by extractability and unique extractability.

5.3 Unlinkability

Unlinkability guarantees that an adversary cannot distinguish standard signatures from adapted pre-signatures, even when the pre-signatures are adapted using adversarially generated witnesses. Dai *et al.* argue that unlinkability is the right security property to guarantee on-chain privacy in the context of atomic swaps [Erw+21].

Definition 14 (Unlinkability). *An adaptor signature scheme* AS *is unlinkable, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* ν *such that for every* $\lambda \in \mathbb{N}$

$$\big|\Pr\big[\mathsf{Unlinkability}_{\mathcal{A},\mathsf{AS}}(\lambda, 0) = 1\big] - \Pr\big[\mathsf{Unlinkability}_{\mathcal{A},\mathsf{AS}}(\lambda, 1) = 1\big]\big| \leq \nu(\lambda) \ ,$$

where experiment $\mathsf{Unlinkability}_{\mathcal{A},\mathsf{AS}}$ *is described in Fig. 10, and the probability is taken over the random choices of all probabilistic algorithms.*

```
Unlinkability_{A,AS}(λ, b)
  1 : $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$
  2 : $b' \leftarrow \mathcal{A}^{\mathsf{Chall}(b,\mathsf{sk},\cdot,\cdot),\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
  3 : return $b'$

Chall(b, sk, m, (Y, y))
  1 : assert $(Y, y) \in \mathsf{Rel}$
  2 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
  3 : $\sigma_0 := \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)$
  4 : $\sigma_1 \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
  5 : return $\sigma_b$

pSign(sk, m, Y)
  1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
  2 : return $\widetilde{\sigma}$

Sign(sk, m)
  1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
  2 : return $\sigma$
```

Figure 10: The security game $\mathsf{Unlinkability}_{\mathcal{A},\mathsf{AS}}(\lambda, b)$.

5.4 Pre-Verify Soundness

We propose *pre-verify soundness* as a new security property for adaptors. Informally, it ensures that the pre-verification algorithm satisfies *computational* soundness w.r.t. the relation Rel. In particular, pVrfy should reject pre-signatures computed using statements $Y \notin \mathsf{Rel}$. 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 complements the property of pre-signature adaptibility (Definition 8), which is restricted to honestly generated pre-signatures on statements in the relation. Pre-verify soundness is important for applications in which the pre-signer also computes the statement w.r.t., which the pre-signature is computed on, since without pre-verify soundness, a malicious signer can trick a verifier into accepting pre-signatures which cannot be adapted even in the presence of a valid witness (c.f. Section 4.2). Suppose the application guarantees that the verifying party always computes the statement. In that case, pre-verify soundness might not be required from the underlying adaptor signature scheme, and more efficient solutions can be used.

Definition 15 (Computational Pre-Verify Soundness)**.** *An adaptor signature scheme* AS *satisfies* computational pre-verify soundness *if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* ν *such that for every* $\lambda \in \mathbb{N}$ *and polynomially-bounded* $Y \notin \mathcal{L}_{\mathsf{Rel}}$*,*

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

For applications where the adversary is allowed to also choose their signing and verification keys, the following stronger statistical notion of pre-verify soundness is useful. The stronger notion is fundamentally similar to the notion of soundness for non-interactive proofs in the plain model . Like non-interactive plain model proofs, the statistical notion of pre-verify soundness is unachievable for *non-trivial* NP relations. For NP relations where checking membership can be done efficiently (e.g., discrete-log language over efficiently recognizable groups), statistically pre-verify soundness can be achieved simply by letting pVrfy algorithm decide membership of the given statement. Nevertheless, we explicitly define this as an additional property for pVrfy as prior constructions of adaptor signatures nor the application they were used in did this check, resulting in a break of the application as discussed in Section 4.

Definition 16 (Statistical Pre-Verify Soundness). *An adaptor signature scheme* AS *satisfies statistical pre-verify soundness if for every* $\lambda \in \mathbb{N}$*, polynomially-bounded* $Y \notin \mathcal{L}_{\mathsf{Rel}}$*, key pair* $(\mathsf{sk}, \mathsf{vk})$ *in the support of* $\mathsf{KGen}(\lambda)$*,*

$$\mathsf{Pr}\left[\exists(m, \widetilde{\sigma}) : \mathsf{pVrfy}(\mathsf{vk}, m, Y) = 1\right] = 0 \ .$$

Pre-Verify Soundness Prevents Non-Adaptable Adaptor Signatures. Our counterexample in Section 4.2 does not achieve pre-verify soundness, since $\perp$ is a valid pre-signature for a statement that is not in the language of the hard relation.

6 Dichotomic Signature Schemes

In this section, we distill a suitable abstraction of signature schemes that can be transformed into adaptor signature schemes without changing the verification algorithm. To this end, we illustrate the algebraic properties with Schnorr signatures [Sch91]; knowing that all other existing adaptor signature schemes can be expressed analogously. We then introduce dichotomic signatures, which are digital signatures characterized by an abstraction of the algebraic structure common to all adaptor signatures. Boneh, Shen, and Waters [BSW06] use a similar abstraction to transform unforgeable signature schemes into strongly unforgeable ones (cf. Lemma 9). In our case, this algebraic structure will help us to construct adaptor signatures from dichotomic signatures in a black-box fashion.

6.1 Motivation — Schnorr Adaptor Signatures

The Schnorr signing process for a message m follows these steps: The signer selects a random value r from the set $\mathbb{Z}_q$ and computes R as $R := g^r$. It calculates a hash value $h := \mathcal{H}(\mathsf{vk}\|R\|m)$ and determines s as $s := r + h \cdot \mathsf{sk}$. The resulting signature is represented as $\sigma := (R, s)$.

We will now explain the modification made to the signing algorithm that results in the computation of a pre-signature. In the context of computing a pre-signature $\widetilde{\sigma}$ for both the message m and the statement $Y = g^y$, the hash computation adjusts to $h := \mathcal{H}(\mathsf{vk}\|R \cdot Y\|m)$, and s remains determined as $r + h \cdot \mathsf{sk}$. The final pre-signature $\widetilde{\sigma}$ takes the form of $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) := (R \cdot Y, s)$. The pre-signature verification algorithm checks if $g^{\widetilde{\sigma}_2} \cdot Y = \widetilde{\sigma}_1 \cdot \mathsf{vk}^h$. To adapt a pre-signature to a full signature, one leverages the homomorphic property of the discrete logarithm relation as follows. Given the witness y one can set $\sigma := (\widetilde{\sigma}_1, \widetilde{\sigma}_2 + y)$. Consequently, given both $\widetilde{\sigma}$ and σ, we can extract the witness y by performing a subtraction.

6.1.1 The Algebraic Properties of Schnorr Adaptor Signatures

We make two observations on the structure of Schnorr adaptor signatures:

- The first observation is that the underlying hard relation $R = \{(Y, y) = (g^y, y)\}$ is an injective homomorphic one-way function. Homomorphism is required for both the verification of the pre-signature and the adaption of a pre-signature to a full signature. Therefore, we abstract this property as $R = \{(Y, y) : Y = \mathsf{OWF}(y)\}$ where OWF is a homomorphic one-way function.

- The second observation is that the signature σ consists of two components $(\sigma_1, \sigma_2) = (R, r + \mathsf{sk} \cdot h)$, which require specific homomorphic properties to guarantee three things simultaneously: first, the verification of the pre-signature succeeds, second, any valid pre-signature can be adapted, and third, the pre-signature can be computed by knowing a statement Y only (and not necessarily the corresponding witness y). These homomorphic properties are the following: Firstly, $\sigma_1 = g^r$ consists of a commitment to the randomness r, whereas $\sigma_2 = r + \mathsf{sk} \cdot h$ contains the (real) randomness r, as well as a hash on the first component σ_1. This structure efficiently computes a pre-signature since modifying σ_1 into $\sigma_1 \cdot Y$ is possible knowing only the statement Y. And since this modified σ_1 is used as input for the computation of σ_2, this modification sufficiently binds Y to the pre-signature. Secondly, a signature is verified by checking if $g^{\sigma_2} = g^{r+\mathsf{sk}\cdot h}$ equals $\sigma_1 \cdot \mathsf{vk}^h = R \cdot \mathsf{vk}^h$. This means, the pre-verification algorithm can multiply g^{σ_2} by Y to implicitly compute $g^{y+r+\mathsf{sk}\cdot h}$ without knowing y. Furthermore, if $(\sigma_1 \cdot Y, g^{\sigma_2} \cdot Y)$ verifies, it means, that $(\sigma_1 \cdot Y, \sigma_2 + y)$ also verifies. This second homomorphic property implies that each adapted pre-signature verifies, guaranteeing pre-signature adaptability.

6.1.2 A Generalization: Dichotomic Signatures

To formalize these intuitive properties, we provide an abstraction of signature schemes that we call *dichotomic signatures*. Our first step is to work with general NP relations that have some homomorphic properties. We consider the relation $\mathsf{Rel} = \{(Y, y) : Y = \mathsf{OWF}(y)\}$, where OWF is a one-way function that is also homomorphic. This abstraction nicely generalizes all prior works that consider specific NP relations with a rich algebraic structure, such as the DLog relation $R = \{(Y, y) : Y = g^y\}$ used for Schnorr signatures [Sch91].

Given this formulation of hard problems, the main challenge is to find a functional abstraction for signatures that supports the main properties of adaptor signatures: pre-signature generation knowing the statement Y and extraction of corresponding witness y given only the pre-signature and the adapted signature. Our analysis of the prior scheme reveals that the signature generation must be a function of the randomness r along with $\mathsf{OWF}(r)$ (where OWF is the function leading to $R = g^r$). Dichotomic signatures formalize this intuition by letting the signature σ be a two-component signature $\sigma := (\sigma_1, \sigma_2)$, which can be computed by the functions Σ_1 and Σ_2, respectively. Here $\sigma_1 \leftarrow \Sigma_1(\mathsf{sk}, m, \mathsf{OWF}(r))$ is a function of the secret key sk, the message m and the image of the randomness $\mathsf{OWF}(r)$; it outputs σ_1. The second function $\sigma_2 \leftarrow \Sigma_2(\mathsf{sk}, m, \sigma_1; r)$ is a function of the secret key, the message, the first component σ_1, and the randomness r; it outputs σ_2. We refer to this splitting in the signature generation process as *decomposability* property of dichotomic signatures. We additionally want a homomorphism property from dichotomic signatures that requires that

$$\Sigma_2(\mathsf{sk}, m, \sigma_1; r) + y = \Sigma_2(\mathsf{sk}, m, \sigma_1; r + y).$$

Looking ahead, this homomorphism will be crucial for pre-signature verification, adaption, and extraction when building adaptor signatures based on dichotomic signatures.

The key idea for a pre-signature generation is to mask σ_1 generation using the statement Y in the following way:

$$\widetilde{\sigma}_1 := \text{Masked}(\sigma_1) \leftarrow \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot Y),$$

while generating $\widetilde{\sigma}_2 \leftarrow \Sigma_2(\mathsf{sk}, m, \widetilde{\sigma}_1; r)$ as usual. We heavily rely on the homomorphic properties of OWF, which is the function used in the relation R and compatible with the signatures' randomness space. Therefore, we can generate a pre-signature using $Y \cdot \mathsf{OWF}(r)$ without knowing the witness y, as required in adaptor signatures. Notice that the pre-signature does not verify as a full signature with respect to the randomness r as we have implicitly set $r + y$ as the randomness for the signature, and we do not know the witness y yet. But we should at least be able to verify if $\widetilde{\sigma}$ is a valid pre-signature generated w.r.t. the correct inputs. To verify this pre-signature $\widetilde{\sigma} := (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$, we again rely on another property of dichotomic signatures, namely, *verifiability*. This property says that there is an algorithm Vrfy' for the dichotomic signature scheme such that

$$\mathsf{Vrfy}(\mathsf{vk}, m, \sigma) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2)).$$

We can now verify the pre-signature, by checking if

$$\mathsf{Vrfy}'(\mathsf{vk}, m, \widetilde{\sigma}_1, \mathsf{OWF}(\widetilde{\sigma}_2) \cdot Y) = 1.$$

Finally, the homomorphism of dichotomic signatures helps us realize the functionality of pre-signature adaption to full signature given the witness y. To see this, given y, we can compute σ_2 as $\sigma_2 := \widetilde{\sigma}_2 + y$ and set the full signature as $\sigma := (\widetilde{\sigma}_1, \sigma_2)$, and it is easy to see that the verification of σ follows. It is also immediate to see the extraction of y given σ and $\widetilde{\sigma}$, as one can compute $y := \sigma_2 - \widetilde{\sigma}_2$.

6.2 Definition of Dichotomic Signatures

Using this intuition of dichotomic signatures, we propose the following formal definition.

Definition 17 (Dichotomic Signature Scheme)**.** *Let* $\mathsf{OWF} : \mathcal{D}_\mathsf{R} \to \mathcal{D}_{\mathsf{R}'}$ *be a homomorphic function. A signature scheme* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *with key space* $\mathcal{D}_\mathsf{K}$*, message-space* $\mathcal{D}_\mathsf{M}$*, randomness space* $\mathcal{D}_\mathsf{R}$*, and signature space* $\mathcal{D}_{\sigma_1} \times \mathcal{D}_\mathsf{R}$ *is a* dichotomic signature w.r.t. the function OWF *if the following holds:*

- *Decomposability. The signature* σ *consists of two parts* $\sigma = (\sigma_1, \sigma_2)$ *that can efficiently be computed by the algorithms* $\Sigma_1 : \mathcal{D}_\mathsf{K} \times \mathcal{D}_\mathsf{M} \times \mathcal{D}_{\mathsf{R}'} \to \mathcal{D}_{\sigma_1}$ *and* $\Sigma_2 : \mathcal{D}_\mathsf{K} \times \mathcal{D}_\mathsf{M} \times \mathcal{D}_{\sigma_1} \times \mathcal{D}_\mathsf{R} \to \mathcal{D}_\mathsf{R}$ *such that* $\Sigma.\mathsf{Sign}(\mathsf{sk}, m; r) = (\sigma_1, \sigma_2) = (\Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r)), \Sigma_2(\mathsf{sk}, m, \sigma_1; r))$.

- *Verifiability. There exists an* DPT *algorithm* $\mathsf{Vrfy}' : \mathcal{D}_\mathsf{K} \times \mathcal{D}_\mathsf{M} \times \mathcal{D}_{\sigma_1} \times \mathcal{D}_{\mathsf{R}'} \to \{0,1\}$ *such that the following holds:*

$$\mathsf{Vrfy}(\mathsf{vk}, m, (\sigma_1, \sigma_2)) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2)) \ .$$

- *Homomorphism. The computation of the algorithm* Σ_2 *is homomorphic in the randomness, that is, for all* $y \in \mathcal{D}_\mathsf{R}$

$$\Sigma_2(\mathsf{sk}, m, \sigma_1; r) + y = \Sigma_2(\mathsf{sk}, m, \sigma_1; r + y) \ .$$

For better readability, we write the homomorphisms with arithmetic operators for additive groups $\mathcal{D}_\mathsf{R}$. Yet, the definition holds for groups that have arbitrary arithmetic operations.

6.3 Constructing Adaptor Signatures from Dichotomic Signatures

We now construct adaptor signatures from dichotomic signatures in a black-box way. In Section 8, we instantiate this generic construction with the BBS^+, CL^+, and Waters^+ signature schemes, yielding the first *natural* adaptor signatures in the standard model.

Construction 1. *Let* $\mathsf{OWF} : \mathcal{D}_\mathsf{R} \to \mathcal{D}_{\mathsf{R}'}$ *be a homomorphic function. Let* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a dichotomic signature scheme with respect to the function* OWF*. Let* Rel *be the canonical hard relation with auxiliary information for* OWF*. We define the adaptor signature scheme* $\mathsf{AS} = (\mathsf{pSign}, \mathsf{Adapt}, \mathsf{pVrfy}, \mathsf{Extract})$ *in Figure 11.*

```
pSign(sk, $m$, $Y$)
1 : if $\mathsf{AuxVrfy}(\mathsf{sk}, Y, \mathsf{aux}) = 0$ return $\bot$
2 : $r \leftarrow\!\!\$\ \mathcal{D}_\mathsf{R}$
3 : $\sigma_1 := \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot Y)$
4 : $\sigma_2 := \Sigma_2(\mathsf{sk}, m, \sigma_1; r)$
5 : return $(\sigma_1, \sigma_2)$

Adapt(vk, $\widetilde{\sigma}$, $y$)
1 : $(\sigma_1, \sigma_2) := \widetilde{\sigma}$
2 : $\widetilde{\sigma_2} = \sigma_2 + y$
3 : return $(\sigma_1, \widetilde{\sigma_2})$

pVrfy(vk, $m$, $Y$, $\widetilde{\sigma}$)
1 : if $\mathsf{StmtVrfy}(\mathsf{vk}, Y, \mathsf{aux}) = 0$ return $0$
2 : $(\sigma_1, \sigma_2) := \widetilde{\sigma}$
3 : return $\mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2) \cdot Y)$

Extract(vk, $\widetilde{\sigma}$, $\sigma$, $Y$)
1 : $(\sigma_1, \sigma_2') := \widetilde{\sigma}$
2 : $(\sigma_1, \sigma_2) := \sigma$
3 : return $\sigma_2 - \sigma_2'$
```

Figure 11: Dichotomic Adaptor Signature Construction. The statement verification (highlighted) is only included if the construction should achieve pre-verify soundness.

We claim the security of Construction 1 in Theorem 4. The formal proof of Theorem 4 requires a novel non-black-box proof technique that we introduce in Section 7. Subsequently, we provide a formal proof of this theorem in Section 8.

Theorem 4. *Let* $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ *be a quasi-injective homomorphic one-way function, and* Rel *be a canonical hard relation with auxiliary input for* OWF*. Let* Σ *be a dichotomic signature scheme with respect to* OWF *and let* AS *be a dichotomic adaptor signature scheme for both* Σ *and* Rel *as per Construction 1. If* Σ *is strongly unforgeable and has a simulatable transparent reduction* $\mathcal{T}$ *from the* $\mathsf{SUF\text{-}CMA}$ *security of* Σ *to an underlying hard problem* Π*, then* AS *achieves full extractability, unique extractability, and unlinkability. If the membership of the statement in the language* $\mathcal{L}_{\mathsf{Rel}}$ *can be checked efficiently, Construction 1 achieves pre-verify soundness.*

7 Transparent Reductions For Signatures

In this section, we introduce a novel *non*-black-box proof technique to prove the security of dichotomic adaptor signatures (c.f. Construction 1). Recall that our goal is the construction of an adaptor signature based on any dichotomic signature scheme. Ideally, we would like to reduce the unforgeability of the adaptor signature scheme to the unforgeability of the signature scheme. However, as we explained in Section 2.3, this is generally impossible, so we developed a new proof technique to show the security of our framework. We call this technique transparent reduction.

7.1 Definition of Transparent Reductions

Intuitively, a transparent reduction follows a natural structure consisting of three parts: an algorithm that computes a (simulated) public key, a signing interface, and a break method (c.f. Fig. 12). The basic idea here is that these interfaces correspond to the expected interfaces of the adversary in the unforgeability game. Thus, the reduction can use the break algorithm on inputs the forgery (m^*, σ^*) of the adversary $\mathcal{A}$, and the internal state of the reduction to compute a solution sol for the hard problem Π. Interestingly, we can exploit this structure non-black-box in subsequent proofs. To do so, the security proof of the adaptor signatures exploits the code of the transparent reduction. The main interesting aspect of this proof technique is that we can show a non-black-box reduction from the full extractability of the adaptor signature scheme to the underlying hard problem of the signature scheme by exploiting the transparent reduction of the given signature scheme.

The invention of this novel proof technique is needed as the simulation of pre-signatures in the full extractability game contradicts the strong unforgeability of the underlying signature scheme: Without the programmability of the RO, the reduction $\mathcal{R}$ against full extractability has only access to a signing oracle and a public key. If we assume that $\mathcal{R}$ can simulate a pre-signature for a message-statement pair (m, Y) chosen by the adversary in this setup, then $\mathcal{R}$ can forge signatures directly: First, $\mathcal{R}$ simulates a pre-signature on a known statement-witness pair (Y, y) and then adapts this pre-signature with a statement y for Y. Note that this adapted pre-signature is computed w.r.t. a different randomness r compared to any full signature output by the signing oracle. Otherwise, the reduction $\mathcal{R}$ is capable of breaking the hardness of the relation by using the extract algorithm on a full signature σ output by the oracle and a simulated pre-signature on the same message m, a statement Y and the same randomness as used in σ. Transparent reductions solve this

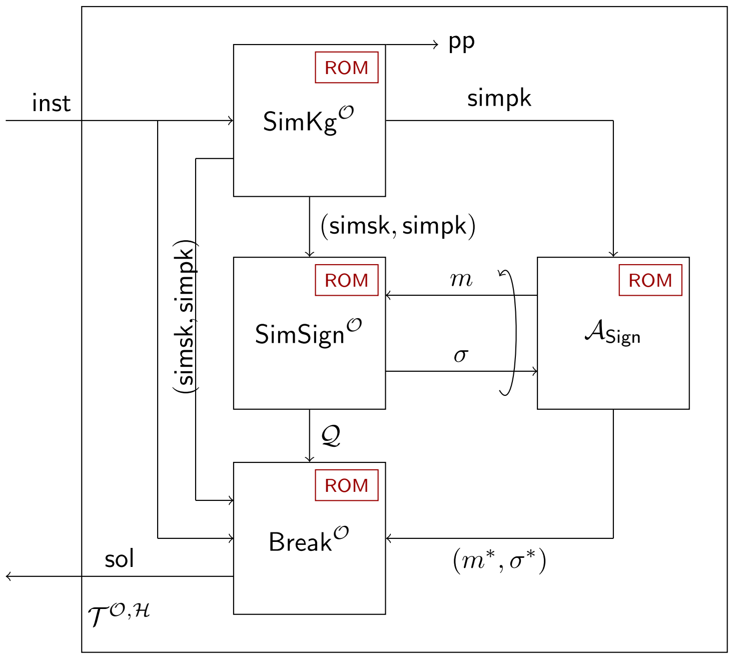
pp
inst
simpk
ROM
$\mathsf{SimKg}^{\mathcal{O}}$
$(\mathsf{simsk}, \mathsf{simpk})$
ROM
$\mathsf{SimSign}^{\mathcal{O}}$
m
σ
ROM
$\mathcal{A}_{\mathsf{Sign}}$
$(\mathsf{simsk}, \mathsf{simpk})$
$\mathcal{Q}$
ROM
$\mathsf{Break}^{\mathcal{O}}$
sol
(m^*, σ^*)
$\mathcal{T}^{\mathcal{O},\mathcal{H}}$

Figure 12: Visualization of a transparent reduction for non-interactive hard problems (in the random oracle model).

issue by providing the reduction against full extractability access to a simulated signing key.

We will define transparent reductions for interactive and non-interactive problems for the standard and the random oracle model below.

Hard (Non-Interactive) Problems. Our formalization of hard non-interactive problems follows [FS10].

Definition 18 (Cryptographic Hard Problem)**.** *A non-interactive (cryptographic) hard problem $\Pi = (I, V)$ consists of two efficient algorithms:*

- $\underline{\mathsf{inst} \leftarrow I(1^\lambda)}$. *The* instance generation *algorithm takes as input the security parameter 1^λ and outputs an instance* inst.
- $\underline{b \leftarrow V(\mathsf{sol}, \mathsf{inst})}$. *The input of the* instance verification *algorithms $V($ sol, $\mathsf{inst})$ is a value* sol *as well as an instance* inst *of a (cryptographic) problem, and outputs a decision bit.*

We call the problem Π interactive *if there exists a helping oracle $\mathcal{O}$ that takes as input a string y and returns an answer x.*

We define the hardness of Π in the following:

Definition 19 (Hardness of a Cryptographic Problem)**.** *Let A (resp. $A^{\mathcal{O}}$) be an efficient algorithm that* solves *the (cryptographic) problem Π if the probability that A (resp. $A^{\mathcal{O}}$) on input $\mathsf{inst} \leftarrow I(1^\lambda)$ outputs sol' such that $V(\mathsf{sol}', \mathsf{inst}) = 1$, is non-negligible. We say that the (non-interactive) (cryptographic) problem Π is* hard *if no efficient algorithm solves it. In terms of concrete security, we say that A runs in time at most t (makes at most Q adaptive queries to $\mathcal{O}$) and solves the hard problem Π with probability ε.*

We provide two examples of hard (cryptographic) problems that are often used for the security of signature schemes as well as the one-more discrete logarithm assumption as an example of an interactive hard problem (see **??**).

Transparent Reductions for (Non-Interactive) Hard Problems. A transparent reduction is divided into several sub-algorithms: a simulated key-generation algorithm SimKg, an interface $\mathsf{SimSign}$ to answer signing queries, and a break interface Break that "converts" the forgery (m^*, σ^*) into a solution of the hard problem Π. If the reduction relies on the random oracle heuristic, then it also stores a list of query and answer pairs. Intuitively, the simulated key-generation algorithm computes a simulated public and private key-pair $(\mathsf{simpk}, \mathsf{simsk})$, which has the property that the public key is indistinguishable from the one of the honest key-generation algorithm. Furthermore, the simulated private key is sufficient to answer signing queries. That is, there exists a possibly stateful algorithm $\mathsf{SimSign}$ that receives simsk as input and computes signatures σ on messages of $\mathcal{A}$'s choice, which verify under the simulated public-key simpk. Finally, the break algorithm receives as input simsk, the query and answer pairs Q of $\mathsf{SimSign}$ oracle invocations, and the forgery $(m^* \sigma^*)$. It outputs the solutions sol for Π.

In the following, we provide a formalization of transparent reductions and show in the next sections that a) the security proofs of many signature schemes have transparent reductions and b) how to make non-black-box use of the transparent reduction in our security proof for adaptor signatures.

Definition 20 (Transparent Reduction)**.** *A transparent reduction* $\mathcal{T} = (\mathsf{SimKg}, \mathsf{SimSign}, \mathsf{Break})$ *for a (non-interactive) hard problem* $\Pi = (I, V)$ *with black-box access to an* PPT *adversary* $\mathcal{A}$ *consists of the following* PPT *algorithms:*

- $\underline{((\mathsf{simsk}, \mathsf{simpk}), \mathsf{pp}) \leftarrow \mathsf{SimKg}(1^\lambda, \mathsf{inst})}$*. The input to the* simulated key-generation *algorithm* $\mathsf{SimKg}(\mathsf{inst})$ *(resp.* $\mathsf{SimKg}^{\mathcal{O}}$*) is the security parameter* 1^λ*, and an instance* inst *of the hard problem* Π*. It outputs a key pair* $(\mathsf{simpk}, \mathsf{simsk})$*, and public parameters* pp*, where* simpk *is a simulated public and* simsk *is a simulated private key.*

- $\underline{\sigma \leftarrow \mathsf{SimSign}(\mathsf{simsk}, m)}$*. The* signing algorithm $\mathsf{SimSign}$ *takes as input a simulated signing key* simsk *and some message* m*, it outputs a signature* σ*. We denote by* $Q_{\mathsf{SimSign}} = \{(m_1, \sigma_1), \ldots, (m_q, \sigma_q)\}$ *the set of all query and answer pairs to* $\mathsf{SimSign}$ *and by* $Q_{\mathcal{O}} = \{(y_1, x_1), \ldots, (y_{q'}, x_{q'})\}$ *the query and answer pairs to the oracle* $\mathcal{O}$*. The same holds for* $\mathsf{SimSign}^{\mathcal{O}}$*.*

- $\underline{\mathsf{sol} \leftarrow \mathsf{Break}(\mathsf{simsk}, Q_{\mathsf{SimSign}}, (m^*, \sigma^*), \mathsf{inst})}$*. The input of the* breaking algorithm Break *(resp.* $\mathsf{Break}^{\mathcal{O}}$*) is a simulated private key* simsk*, the set of query/answer pairs to the sign interface* $Q_{\mathsf{SimSign}} = \{(m_1, \sigma_1), \ldots, (m_q, \sigma_q)\}$ *(and the set* $Q_{\mathcal{O}} = \{(y_1, x_1), \ldots, (y_{q'}, x_{q'})\}$ *of queries to* $\mathcal{O}$*), a putative forgery* (m^*, σ^*)*, and an instance* inst *of the hard problem* Π*. It returns a putative solution* sol*.*

The correctness of transparent reductions for non-interactive hard problems is defined as follows. The definition for the interactive case follows analogously.

Definition 21 (Correctness of Transparent Reductions)**.** *We say that the transparent reduction* $\mathcal{T}_{\Pi}^{\mathcal{A}}$ *is* correct *if the following holds: Let* $\mathsf{inst} \leftarrow I(1^\lambda)$ *be an instance of the hard problem* Π*,* $(\mathsf{simsk}, \mathsf{simpk}) \leftarrow \mathsf{SimKg}(1^\lambda, \mathsf{inst})$ *be the simulated key pair,* $Q_{\mathsf{SimSign}} = \{(m_1, \sigma_1), \ldots, (m_q, \sigma_q)\}$ *the set of query/answer pair of* $\mathcal{A}(\mathsf{simpk})$ *to* $\mathsf{SimSign}(\mathsf{simsk}, \cdot)$*, and let* (m^*, σ^*) *the final output of* $\mathcal{A}$*. If* $(m^*, q^*) \notin Q_{\mathsf{SimSign}}$ *and* $\mathsf{Vrfy}(\mathsf{simpk}, m^*, \sigma^*) = 1$*, then we call a reduction* correct *if and only if* $\Pr[V(\mathsf{sol}', \mathsf{inst}) = 1] \geq \varepsilon$*, where* $\mathsf{sol}' \leftarrow \mathsf{Break}(\mathsf{simsk}, Q_{\mathsf{SimSign}}, (m^*, \sigma^*), \mathsf{inst})$*,* ε *is non-negligible, and the probability is taken over the random choices of* V*,* Break*,* SimKg *and* $\mathsf{SimSign}$*.*

Transparent Reductions and the Random Oracle Model. Our formalization of transparent reductions naturally carries over to reductions in the random oracle model [BR93] that we define as follows.

Definition 22. *A* transparent reduction $\mathcal{T} = (\mathsf{SimKg}^{\mathsf{H}}, \mathsf{SimSign}^{\mathsf{H}}, \mathsf{Break}^{\mathsf{H}})$ *in the random oracle model for some hard problem* $\Pi = (I, V)$ *with black-box access to an efficient adversary* $\mathcal{A}$ *consists of efficient algorithms* $(\mathsf{SimKg}^{\mathsf{H}}, \mathsf{SimSign}^{\mathsf{H}}, \mathsf{Break}^{\mathsf{H}})$ *which are defined analogously to Definition 20, and* H *is defined as follows:*

- $\underline{\mathsf{H}.}$ *The reduction stores a list* Q_{H} *of input and output pairs* (a, b)*, meaning that* $\mathsf{H}(a) = b$*. The interface* H *provides two modes:* $\mathsf{H}(\mathtt{get}, a)$ *returns the value* b *if* $(a, b) \in Q_{\mathsf{H}}$ *and* $\perp$*, otherwise. The mode* $\mathsf{H}(\mathtt{set}, (a, b))$ *adds the entry* (a, b) *to* Q_{H} *iff* $(a, \cdot) \notin Q_{\mathsf{H}}$*.*

We assume that all algorithms have access to Q_{H}*.*

7.2 Simulatable Transparent Reductions for Dichotomic Signatures

To prove the security of the resulting adaptor signatures, the underlying dichotomic signature scheme needs to be strongly unforgeable. Furthermore, we need a transparent reduction $\mathcal{T}$ from the strong unforgeability to an underlying hard problem with an additional property, which we call *simulatability*. Looking ahead, simulatability will allow our security reduction to compute values that are indistinguishable from pre-signatures for any efficient distinguisher by having as input a simulated secret key simsk rather than a signing key sk. Even if pre-signatures are not in the scope of signature schemes, we can use the structure of dichotomic signatures to define the simulatability notion by using the functions Σ_1 and Σ_2.

Definition 23 (Simulatable Transparent Reductions for Dichotomic Signatures)**.** *Let* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a dichotomic signature scheme based on the function* $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ *and* $\mathcal{T} = (\mathsf{SimKg}, \mathsf{SimSign}, \mathsf{Break})$ *be a transparent reduction from the* $\mathsf{SUF\text{-}CMA}$ *security of* Σ *to an underlying hard problem* Π*. We refer to* $\mathcal{T}$ *as a* simulatable *transparent reduction for a canonical hard relation of* OWF *if the following holds: There is an efficient algorithm* Sim *that on input a simulated secret key* $\mathsf{simsk} \leftarrow \mathsf{SimKg}(\mathsf{inst})$*, a message* $m \in \mathcal{D}_{\mathsf{M}}$ *and a image* $Y \in \mathcal{D}_{\mathsf{R}'}$ *of the homomorphic function* OWF *outputs a value* $\widetilde{\sigma}$*, such that for any* PPT *distinguisher* $\mathcal{D}$ *and for each hard instance* $\mathsf{inst} \in \Pi$

$$|\Pr\big[\mathsf{Exp}_{\mathcal{D}}^{1}(\mathsf{inst}, 1^\lambda) = 1\big] - \Pr\big[\mathsf{Exp}_{\mathcal{D}}^{0}(\mathsf{inst}, 1^\lambda) = 1\big]| \leq \nu(\lambda)$$

```
$\mathsf{Exp}^0_{\mathcal{D}}(1^\lambda)$
    1 : $(\mathsf{sk}, \mathsf{vk}) \leftarrow \Sigma.\mathsf{KGen}(1^\lambda)$
    2 : $b' \leftarrow \mathcal{D}^{\mathcal{O}_0(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
    3 : $\textbf{return } b' == 0$

$\mathcal{O}_0(\mathsf{sk}, m, Y)$
    1 : $r \leftarrow\$ \mathcal{D}_{\mathsf{R}}$
    2 : $\sigma_1 := \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot Y)$
    3 : $\sigma_2 := \Sigma_2(\mathsf{sk}, m, \sigma_1; r)$
    4 : $\textbf{return } (\sigma_1, \sigma_2)$

$\mathsf{Exp}^1_{\mathcal{D}}(1^\lambda, \mathsf{inst})$
    1 : $(\mathsf{simsk}, \mathsf{simpk}) \leftarrow \mathcal{T}.\mathsf{SimKg}(1^\lambda, \mathsf{inst})$
    2 : $b' \leftarrow \mathcal{D}^{\mathcal{O}_1(\mathsf{simsk},\cdot,\cdot)}(\mathsf{simpk})$
    3 : $\textbf{return } b' == 1$

$\mathcal{O}_1(\mathsf{simsk}, m, Y)$
    1 : $\textbf{return } \mathsf{Sim}(\mathsf{simsk}, m, Y)$
```

Figure 13: The simulatability game for transparent reductions.

holds, where Fig. 13 shows the corresponding games and the probability is taken over the random choices of all probabilistic algorithms.

Simulatable transparent reductions provide an additional oracle compared to the oracles used in signature schemes, namely the pre-signature oracle. The adversary having access to this additional oracle raises the question of which signatures are suitable inputs for the transparent reduction's breaking algorithm. To answer this question, we define *fresh signatures.* Simply put, we consider a signature fresh if it has not been obtained by an oracle or using trivial transformations on oracle outputs. More formally:

Definition 24 (Fresh Signature)**.** *A signature σ is fresh if it is not output by an oracle. If a pre-signature $\widetilde{\sigma} = (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ for a statement Y is output by an oracle, we mark each signature $\sigma^* = (\sigma^*_1, \sigma^*_2)$ as non-fresh that satisfies $\sigma^*_1 = \widetilde{\sigma}_1$ and $\sigma^*_2 - \widetilde{\sigma}_2 = y'$ such that $\mathsf{OWF}(y') = Y$.*

8 Secure Dichotomic Adaptor Signatures

In this section, we show the security of dichotomic adaptor signatures (c.f. Fig. 11) by proving extractability, unique extractability, unlinkability, and pre-verify soundness. Afterward, we show how to build adaptor signatures from BBS^+, CL^+, and Waters^+. To show the broad applicability of our framework, we show that our framework also covers ID-based signature schemes, which a previous work showed w.r.t. a weaker security model in [Aum+21].

8.1 Proof of Security

We now show full extractability, unlinkability, unique extractability, and pre-verify soundness in separate lemmas. We postpone the correctness proofs to Appendix C.

Lemma 1 (Extractability)**.** *Let $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ be a quasi-injective homomorphic one-way function, and Rel be a canonical hard relation with auxiliary input for OWF. Let Σ be* *a dichotomic signature scheme with respect to* OWF *and let* AS *be a dichotomic adaptor signature scheme for both* Σ *and* Rel *as per Fig. 11. If* Σ *is* $\mathsf{SUF\text{-}CMA}$ *secure and has a simulatable transparent reduction* $\mathcal{T}$ *from the* $\mathsf{SUF\text{-}CMA}$ *security of* Σ *to an underlying hard problem* Π*, then* AS *satisfies extractability (c.f. Definition 12).*

Proof. Before we start with our proof, we first provide a high-level overview of how we build a reduction from the extractability of AS to the hardness of the underlying hard problem Π using the algorithms provided by the simulatable transparent reduction $\mathcal{T}$. The reduction $\mathcal{R}$ has as input the instance inst and simulates the game $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}$ to the adversary $\mathcal{A}$ the following way. To provide a public key and a signing oracle to $\mathcal{A}$, the reduction $\mathcal{R}$ runs the algorithms SimKg and $\mathsf{SimSign}$, which are provided by the transparent reduction $\mathcal{T}$. To provide a pre-signing oracle, $\mathcal{R}$ uses the simulatability property of $\mathcal{T}$. Having access to these oracles, the adversary $\mathcal{A}$ eventually outputs a valid forgery, i.e., a message-signature pair (m^*, σ^*). Upon receiving such a forgery, the reduction $\mathcal{R}$ runs the Break algorithm provided by the transparent reduction to compute a solution of the inst.

Using this intuition, we now start the proof and show in a series of game hops that we can indeed simulate the game Ext to $\mathcal{A}$ in an indistinguishable fashion using the algorithms provided by the simulatable transparent reduction. Our game hops range from the game $\mathcal{G}_0$, which equals the original Ext game, to the game $\mathcal{G}_3$, in which each valid forgery provided by $\mathcal{A}$ can be used as input to the Break algorithm, such that Break computes a solution for the instance inst.

Game $\mathcal{G}_0$: The first game $\mathcal{G}_0$ is the original $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}$ game. Since $\mathcal{G}_0$ simulates the $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}$ game perfectly, it holds that $\Pr[\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}(\lambda) = 1] = \Pr[\mathcal{G}_0(\lambda) = 1]$. A formal description of game $\mathcal{G}_0$ is given in Fig. 14.

Game $\mathcal{G}_1$: The second game $\mathcal{G}_1$ only differs from game $\mathcal{G}_0$ when it aborts in Line 5. We call this event $\mathsf{Break}_{\mathsf{Rel}}$. At an intuitive level, $\mathsf{Break}_{\mathsf{Rel}}$ occurs if the forgery of the adversary allows breaking a challenge instance of the hard relation. This game hop does not differ from $\mathsf{Ext}_{\mathcal{A},\mathsf{AS}}$ if the adversary returns an adapted pre-signature on a non-challenge statement since the winning condition covers this already. It holds that

$$\Pr[\mathcal{G}_1(\lambda) = 1] = \Pr[\mathcal{G}_1(\lambda) = 1 \wedge \mathsf{Break}_{\mathsf{Rel}}] + \Pr[\mathcal{G}_1(\lambda) = 1 \wedge \neg\mathsf{Break}_{\mathsf{Rel}}].$$

A formal description of game $\mathcal{G}_1$ is given in Fig. 14.

Claim 1. *Let* $\mathsf{Break}_{\mathsf{Rel}}$ *be the event, whereby* $\mathcal{G}_1$ *aborts in Line 5. Then, the adversary* $\mathcal{A}$ *found a signature forgery* σ^* *that allows extracting a challenge witness using a previously output pre-signature. If* $\mathsf{Break}_{\mathsf{Rel}}$ *happens, then we can build a reduction* $\mathcal{R}$ *that can break the hardness of the hard relation* Rel*. Thus,* $\Pr[\mathcal{G}_1(\lambda) = 1 \wedge \mathsf{Break}_{\mathsf{Rel}}]$ *occurs only with negligible probability* $\nu_1(1^\lambda)$*.*

Proof of Claim 1. We prove Claim 1 with a reduction $\mathcal{R}_{\mathsf{Rel}}$ that uses $\mathcal{A}$ from game $\mathcal{G}_1$ to break the hardness of the relation Rel if $\mathsf{Break}_{\mathsf{Rel}}$ happens. When $\mathsf{Break}_{\mathsf{Rel}}$ happens, then the forgery of $\mathcal{A}$ allows extracting a valid witness for a challenge statement $Y \in \mathcal{Q}_{\mathsf{stmt}}$. Such a forgery breaks the hardness of Rel since the challenge statements are honestly computed, and the corresponding witnesses are not forwarded either to $\mathcal{A}$ or to the

```
$\mathcal{G}_0(\lambda)$ $\mathcal{G}_1(\lambda)$
1 : $\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}}, \mathcal{Q}_{\mathsf{stmt}} := \emptyset, (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot),\mathsf{NewY}(\lambda)}(\mathsf{vk})$
3 : if $\exists (Y, \widetilde{\sigma}) \in \mathcal{Q}_{\mathsf{pSign}}[m^*]$ s.t. $Y \in \mathcal{Q}_{\mathsf{stmt}}$
4 :  $\wedge (Y, \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma^*, Y)) \in \mathsf{Rel}$ then
5 :    abort
6 : assert $\mathsf{Vrfy}(\mathsf{vk}, m, \sigma^*)$
7 : assert $m^* \notin \mathcal{Q}_{\mathsf{Sign}}$
8 : return $\forall (Y, \widetilde{\sigma}) \in \mathcal{Q}_{\mathsf{pSign}}[m^*] : (Y, \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma^*)) \notin \mathsf{Rel}$

$\mathsf{NewY}(\lambda)$
1 : $(Y, y) \leftarrow \mathsf{Rel.RGen}(1^\lambda)$; $\mathcal{Q}_{\mathsf{stmt}} \overset{\cup}{\leftarrow} Y$; return $Y$

$\mathsf{Sign}(\mathsf{sk}, m)$
1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
2 : $\mathcal{Q}_{\mathsf{Sign}} \overset{\cup}{\leftarrow} \{m\}$
3 : return $\sigma$

$\mathsf{pSign}(\mathsf{sk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
2 : return $\widetilde{\sigma}$
```

Figure 14: Game $\mathcal{G}_0(\lambda)$ and Game $\mathcal{G}_1(\lambda)$ (highlighted parts in gray).

oracles provided to $\mathcal{A}$. In addition, we can break the hardness of Rel even if $\mathcal{A}$ does not win the game $\mathcal{G}_0$ after the event $\mathsf{Break}_{\mathsf{Rel}}$ happens, since the event already covers that the extracted witness is valid. Upon input of a challenge statement Y^*, the reduction $\mathcal{R}_{\mathsf{Rel}}$ computes a public and a secret key $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(Y^*)$. We can assume that the public key vk aligns with the challenge statement Y^* since the public parameters of both algorithms can be encoded to the auxiliary information of Y^*. The reduction $\mathcal{R}_{\mathsf{Rel}}$ provides vk to $\mathcal{A}$ and answers the pSign and Sign oracle queries of $\mathcal{A}$ using the signing key sk. To answer the i-th query of the NewY oracle, the reduction first samples a random element $r_i \leftarrow\!\!\$\ \mathcal{D}_{\mathsf{R}}$, stores r_i and outputs $Y_i := \mathsf{OWF}(r_i) \cdot Y^*$. Since the one-way function OWF is homomorphic and r_i is sampled uniformly at random, the statement Y_i has the right distribution. Eventually, $\mathcal{A}$ outputs a forgery (m^*, σ^*). If the event $\mathsf{Break}_{\mathsf{Rel}}$ occurs with non-negligible probability (which we assume in Claim 1), there exists a pre-signature $\widetilde{\sigma}_i$ on a statement Y_i, such that $y := \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma^*, Y_i)$ is a valid witness for Y_i. Since the reduction $\mathcal{R}_{\mathsf{Rel}}$ stores the rerandomization factor r_i, it can compute $y^* = y - r_i$ as a valid witness for the challenge statement Y^*. This reduction breaks the hardness of the hard relation Rel, and thus, the event $\mathsf{Break}_{\mathsf{Rel}}$ happens only with negligible probability. □

Game $\mathcal{G}_2$: In this game, we replace the key generation algorithm, the signing algorithm, and the pre-signing algorithm with the corresponding simulated algorithms provided by the simulatable transparent reduction $\mathcal{T}$. We show game $\mathcal{G}_2$ in Figure 15.

Claim 2. *By the simulatability of $\mathcal{T}$, the difference between the games $\mathcal{G}_1$ and $\mathcal{G}_2$ is negligible. This means,* $\Pr[\mathcal{G}_1(\lambda) = 1] \leq \Pr[\mathcal{G}_2(\lambda) = 1] + \nu_2(1^\lambda)$.

Proof of Claim 2. We assume, by contradiction, that the difference between game $\mathcal{G}_1$ and game $\mathcal{G}_2$ is non-negligible. I.e., there exists a distinguisher $\mathcal{A}$, that discovers the gap between the games $\mathcal{G}_1$ and $\mathcal{G}_2$, with non-negligible probability ϵ. We then build a reduction $\mathcal{R}$ that breaks the simulatability of the transparent reduction of Σ. The reduction $\mathcal{R}$ obtains as input a verification key vk, where vk is either generated by KGen

```
$\mathcal{G}_2(\lambda, \mathsf{inst})$
1 : $\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}} := \emptyset$
2 : $(\mathsf{simsk}, \mathsf{simpk}) \leftarrow \mathcal{T}.\mathsf{SimKg}(\mathsf{inst})$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{simsk},\cdot),\mathsf{pSign}(\mathsf{simsk},\cdot,\cdot),\mathsf{NewY}(\lambda)}(\mathsf{simpk})$
4 : if $\exists (Y, \widetilde{\sigma}) \in \mathcal{Q}_{\mathsf{pSign}}[m^*]$ s.t. $Y \in \mathcal{Q}_{\mathsf{stmt}}$
5 :  $\wedge (Y, \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma^*, Y)) \in \mathsf{Rel}$
6 :     abort
7 : assert $\mathsf{Vrfy}(\mathsf{simpk}, m, \sigma^*)$
8 : assert $m^* \notin \mathcal{Q}_{\mathsf{Sign}}$
9 : return $\forall (Y, \widetilde{\sigma}) \in \mathcal{Q}_{\mathsf{pSign}}[m^*] : (Y, \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma^*)) \notin \mathsf{Rel}$

$\mathsf{NewY}(\lambda)$
1 : $(Y, y) \leftarrow \mathsf{RGen}(1^\lambda)$
2 : return $Y$

$\mathsf{Sign}(\mathsf{simsk}, m)$
1 : $\sigma \leftarrow \mathcal{T}.\mathsf{SimSign}(\mathsf{simsk}, m)$
2 : $\mathcal{Q}_{\mathsf{Sign}} \overset{\cup}{\leftarrow} \{m\}$
3 : return $\sigma$

$\mathsf{pSign}(\mathsf{simsk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathcal{T}.\mathsf{Sim}(\mathsf{simsk}, m, Y)$
2 : $\mathcal{Q}_{\mathsf{pSign}}[m] \overset{\cup}{\leftarrow} \{Y, \widetilde{\sigma}\}$
3 : return $\widetilde{\sigma}$
```

Figure 15: The game $\mathcal{G}_2(\lambda)$.

or by $\mathcal{T}.\mathsf{SimKg}$. Furthermore, $\mathcal{R}$ has access to an oracle $\mathcal{O}$. $\mathcal{R}$ provides a pre-sign oracle to its adversary by forwarding the request (m, Y) of the adversary to its oracle. The sign oracle is simulated by requesting $(m, \mathsf{OWF}(0))$ to $\mathcal{O}$. Since OWF is homomorphic and quasi-injective, $\mathsf{OWF}(0)$ is the identity in the group $\mathcal{D}_{\mathsf{R}'}$, and hence the output $(\sigma_1, \sigma_2) := \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot \mathsf{OWF}(0)), \Sigma_2(\mathsf{sk}, m, \sigma_1; r) = \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r)), \Sigma_2(\mathsf{sk}, m, \sigma_1; r)$ which is a normal signature on the message m. The NewY oracle has only public inputs and hence is simulated trivially. If the bit b equals 0, then the reduction $\mathcal{R}$ simulated game $\mathcal{G}_1$ perfectly. Else, the reduction perfectly simulates game $\mathcal{G}_2$. Using this observation, we are now in a situation where there exists a distinguisher $\mathcal{A}$ that discovers the gap between $\mathcal{G}_1$ and $\mathcal{G}_2$ with non-negligible probability ϵ, and where the games $\mathcal{G}_1$ and $\mathcal{G}_2$ perfectly match the experiments Exp^0 and Exp^1 from the simulatable of the transparent reduction. Therefore, our reduction $\mathcal{R}$ can leverage this distinguisher to break the simulatability of $\mathcal{T}$ with the same non-negligible probability ϵ by forwarding the distinguishers output. □

Game $\mathcal{G}_3$: In this game, we output a solution to the instance inst of the hard problem if the adversary outputs a valid forgery. Game $\mathcal{G}_3$ is depicted in Fig. 16. Between the games $\mathcal{G}_2$ and $\mathcal{G}_3$, there is no difference in the view of the adversary. Yet, the game $\mathcal{G}_3$ is a reduction from the strong unforgeability of Σ to its underlying hard problem.

Claim 3. *If the adversary outputs a valid message-forgery pair (m^*, σ^*) for $\mathcal{G}_2$, this message-forgery pair allows the transparent reductions breaking algorithm $\mathcal{T}.\mathsf{Break}$ to compute a solution for the instance* inst *of the underlying hard problem.*

Proof of Claim 3. Alongside the simulated secret key simsk, the queue $\mathcal{Q}$, and the in-

```
$\mathcal{G}_3(\lambda, \mathsf{inst})$
1 : $\mathcal{Q}_{\mathsf{Sign}}, \mathcal{Q}_{\mathsf{pSign}}, \mathcal{Q}_{\mathsf{stmt}} := \emptyset$
2 : $(\mathsf{simsk}, \mathsf{simpk}) \leftarrow \mathcal{T}.\mathsf{SimKg}(\mathsf{inst})$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{simsk},\cdot),\mathsf{pSign}(\mathsf{simsk},\cdot,\cdot),\mathsf{NewY}(\lambda)}(\mathsf{simpk})$
4 : if $\exists (Y, \widetilde{\sigma}) \in \mathcal{Q}_{\mathsf{pSign}}[m^*]$ s.t. $Y \in \mathcal{Q}_{\mathsf{stmt}}$
5 :  $\wedge (Y, \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma^*, Y)) \in \mathsf{Rel}$
6 :    abort
7 : assert $\mathsf{Vrfy}(\mathsf{simpk}, m, \sigma^*)$
8 : assert $m^* \notin \mathcal{Q}_{\mathsf{Sign}}$
9 : return $\mathcal{T}.\mathsf{Break}(\mathsf{simsk}, \mathcal{Q}_{\mathsf{Sign}} \cup \mathcal{Q}_{\mathsf{pSign}}, (m^*, \sigma^*), \mathsf{inst})$

$\mathsf{NewY}(\lambda)$
1 : $(Y, y) \leftarrow \mathsf{RGen}(1^\lambda)$
2 : return $Y$

$\mathsf{Sign}(\mathsf{simsk}, m)$
1 : $\sigma \leftarrow \mathcal{T}.\mathsf{SimSign}(\mathsf{simsk}, m)$
2 : $\mathcal{Q}_{\mathsf{Sign}} \overset{\cup}{\leftarrow} \{m\}$
3 : return $\sigma$

$\mathsf{pSign}(\mathsf{simsk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathcal{T}.\mathsf{Sim}(\mathsf{simsk}, m, Y)$
2 : $\mathcal{Q}_{\mathsf{pSign}}[m] \overset{\cup}{\leftarrow} \{Y, \widetilde{\sigma}\}$
3 : return $\widetilde{\sigma}$
```

Figure 16: The game $\mathcal{G}_3(\lambda, \mathsf{inst})$.

stance of the hard problem inst, the breaking algorithm expects a message-signature forgery pair (m^*, σ^*) containing a fresh signature (meaning that was never returned by the signing oracle before). This means, $(m^*, \sigma^*) \notin \mathcal{Q}_{\mathsf{Sign}}$. Since we simulate the pre-signing oracle using the simulated secret key, we add all messages that were asked to the pSign oracle to this queue. According to Lemma 22, a forgery signature on a message m is fresh if it does not extract with any pre-signature on m. Since (m^*, σ^*) is a valid forgery for game $\mathcal{G}_2$, by the winning condition of $\mathcal{G}_2$ it is also fresh. Consequently, the message-signature pair (m^*, σ^*) is a valid forgery for the StrongSigForge game; thus, the forgery is a sufficient input for $\mathcal{T}.\mathsf{Break}$. □

We now use Claim 3 to conclude our proof. As $\mathcal{G}_3$ simulates StrongSigForge perfectly, if it does not abort, we now have:

$$\Pr[\mathsf{Ext}(\lambda) = 1] \leq \Pr[\mathsf{StrongSigForge}(\lambda) = 1] + \nu_1(1^\lambda) + \nu_2(1^\lambda).$$

We assumed that $\mathcal{A}$ wins Ext with non-negligible probability, and thus $\mathcal{A}$ also wins StrongSigForge with non-negligible probability. This contradicts the SUF-CMA security of Σ; thus, such an adversary can not exist. □

Lemma 2 (Unique Extractability)**.** *Let* OWF : $\mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R'}}$ *be a quasi-injective homomorphic one-way function, and* Rel *be a canonical hard relation with auxiliary input for* OWF*. Let* Σ *be a dichotomic signature scheme with respect to* OWF *and let* AS *be a dichotomic adaptor signature scheme for both* Σ *and* Rel *as per Construction 1. If* Σ *is* SUF-CMA *secure and has a simulatable transparent reduction* $\mathcal{T}$ *from the* SUF-CMA *security of* Σ *to an underlying hard problem* Π*, then* AS *satisfies unique extractability (c.f. Definition 13).*

Proof of Lemma 2. The proof of this lemma follows the structure of Lemma 1. By contradiction, we use the ability of any adversary breaking unique extractability to construct a novel adversary that can break the hardness of the underlying hardness assumption of the signature scheme Σ. To do so, we define a series of game hops where the first game, $\mathcal{G}_0$ is the original $\mathsf{UniqueExtractability}$ game, and the last game allows breaking the underlying hard instance of the strong unforgeability of Σ.

Game $\mathcal{G}_0$: The first game $\mathcal{G}_0$ is the original $\mathsf{UniqueExtractability}$ game. Since $\mathcal{G}_0$ perfectly simulates the game $\mathsf{UniqueExtractability}$, it holds that $\Pr[\mathsf{UniqueExtractability}(\lambda) = 1] = \Pr[\mathcal{G}_0(\lambda) = 1]$. A formal description of the game $\mathcal{G}_0$ is given in Fig. 17.

```
G_0(λ)
  1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
  2 : $(m, Y, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathsf{pSign}(\mathsf{sk},\cdot,\cdot),\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$
  3 : assert $(\sigma \neq \sigma') \wedge \mathsf{Vrfy}(\mathsf{vk}, m, \sigma) \wedge \mathsf{Vrfy}(\mathsf{vk}, m, \sigma')$
  4 : assert $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma})$
  5 : $y \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma')$
  6 : return $(Y, y) \in \mathsf{Rel} \wedge (Y, y') \in \mathsf{Rel})$

Sign(sk, m)
  1 : $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathsf{sk}, m)$
  2 : return $\sigma$

pSign(sk, m, Y)
  1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{sk}, m, Y)$
  2 : return $\widetilde{\sigma}$
```

Figure 17: The first game $\mathcal{G}_0(\lambda)$.

Game $\mathcal{G}_1$: In the second game, we replace the key generation algorithm, the signing algorithm, and the pre-signing algorithm with the corresponding simulated algorithms provided by the simulatable transparent reduction $\mathcal{T}$. By the simulatability of $\mathcal{T}$, the

```
G_1(λ, inst)
  1 : $(\mathsf{simpk}, \mathsf{simsk}) \leftarrow \mathsf{SimKg}(\mathsf{inst})$
  2 : $(m, Y, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathsf{pSign}(\mathsf{simsk},\cdot,\cdot),\mathsf{Sign}(\mathsf{simsk},\cdot)}(\mathsf{simpk})$
  3 : assert $(\sigma \neq \sigma') \wedge \mathsf{Vrfy}(\mathsf{simpk}, m, \sigma) \wedge \mathsf{Vrfy}(\mathsf{simpk}, m, \sigma')$
  4 : assert $\mathsf{pVrfy}(\mathsf{simpk}, m, Y, \widetilde{\sigma})$
  5 : $y \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma')$
  6 : return $(Y, y) \in \mathsf{Rel} \wedge (Y, y') \in \mathsf{Rel})$

pSign(simsk, m, Y)
  1 : $\widetilde{\sigma} \leftarrow \mathcal{T}.\mathsf{SimPSign}(\mathsf{simsk}, m, Y)$
  2 : return $\widetilde{\sigma}$

Sign(simsk, m)
  1 : $\sigma \leftarrow \mathcal{T}.\mathsf{SimSign}(\mathsf{simsk}, m)$
  2 : return $\sigma$
```

Figure 18: The second game $\mathcal{G}_1(\lambda)$.

difference between the games is negligible (c.f. Proof of Claim 2) and we have $\Pr[\mathcal{G}_0 = 1] \leq \Pr[\mathcal{G}_1 = 1] + \nu_1(1^\lambda)$. We provide a formal description of the game $\mathcal{G}_1$ in Fig. 18.

Claim 4. *If the adversary wins the game $\mathcal{G}_1$ with non-negligible probability, the adversaries output $(m, Y, \widetilde{\sigma}, \sigma, \sigma')$ contains a fresh signature with non-negligible probability.*

Proof of Claim 4. To prove the claim, we first observe that the pre-signature and two signatures must follow a certain structure to fulfill the winning condition of game $\mathcal{G}_1$ (the winning condition of unique extractability): The winning condition enforces, that

$$y \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma') \text{ and } (Y, y) \in \mathsf{Rel} \wedge (Y, y') \in \mathsf{Rel}$$

hold. This fact allows two observations. First, since the relation of Construction 1 is quasi-injective, we know that $y = y'$ with overwhelming probability. Second, the extract algorithm $\mathsf{Extract}$ on input a pre-signature $\widetilde{\sigma} = (\widetilde{\sigma_1}, \widetilde{\sigma_2})$ and a signature $\sigma = (\sigma_1, \sigma_2)$ of Construction 1 computes $y = \sigma_2 - \widetilde{\sigma_2}$. Since $y = y'$ with overwhelming probability, this implies that with overwhelming probability

$$\sigma_2 = \widetilde{\sigma_2} + y = \sigma_2'.$$

Next, we show that it is negligible that the oracles of the reduction output at most two values that have the same second component $(\cdot, \sigma_2)$. Each time $\mathcal{A}$ invokes an oracle, the oracle first samples a fresh random value r and computes the functions Σ_1 and Σ_2 afterward. The probability that two invocations of the oracle sample have the same random value r is negligible. Yet, if the oracle does not sample the same random value r, the output of the oracle differs in the second component σ_2 since the dichotomic signature has a homomorphic property requiring

$$\Sigma_2(\mathsf{sk}, m, r + x) = \Sigma_2(\mathsf{sk}, m, r) + x.$$

Henceforth, if $r \neq r'$, we have $r = r' + x$ for a non-zero x, which leads to

$$\Sigma_2(\mathsf{sk}, m, r') = \Sigma_2(\mathsf{sk}, m, r + x) = \Sigma_2(\mathsf{sk}, m, r) + x \neq \Sigma_2(\mathsf{sk}, m, r),$$

since Σ_2 is a deterministic function. Therefore, the probability that $\sigma_2 = \sigma_2'$ and both values are output by the oracles of the reduction is negligible.

Putting things together, the adversary has to output three values that share the same second component σ_2 (or $\sigma_2 + y$ in case of the pre-signature) but are unequal. While it is possible to output two signatures that share the same second component - via adapting - and were output by oracles before, we have shown that it is negligible that an oracle outputs a third such signature. Therefore, at least one of the three output values yields a fresh signature. □

Game $\mathcal{G}_2$: In the third game, we replace the output of $\mathcal{G}_1$ by the break algorithm of $\mathcal{T}$. Due to Claim 4, this $\mathcal{G}_2$ finds a solution for inst, whenever the adversary outputs a valid forgery. Game $\mathcal{G}_2$ simulates $\mathsf{StrongSigForge}$ perfectly and the view of $\mathcal{A}$ does not change between the games $\mathcal{G}_1$ and $\mathcal{G}_2$, hence $\Pr[\mathcal{G}_1] = 1 = \Pr[\mathcal{G}_2] = 1$. Game $\mathcal{G}_2$ is depicted in Fig. 19. This series of game hops concludes the proof. As $\mathcal{G}_2$ simulates $\mathsf{StrongSigForge}$ perfectly, it holds:

$$\Pr[\mathsf{UniqueExtractability}(\lambda) = 1] \leq \Pr[\mathsf{StrongSigForge}(\lambda) = 1] + \nu_1(1^\lambda).$$

We assumed by contradiction that $\mathcal{A}$ wins $\mathsf{UniqueExtractability}$ with non-negligible probability, and thus $\mathcal{A}$ wins $\mathsf{StrongSigForge}$ with non-negligible probability. This contradicts the $\mathsf{SUF\text{-}CMA}$ security of Σ. Hence, no such an adversary can exist. □

```
$\mathcal{G}_2(\lambda, \mathsf{inst})$
1 : $(\mathsf{simpk}, \mathsf{simsk}) \leftarrow \mathsf{SimKg}(\mathsf{inst})$
2 : $(m, Y, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathsf{pSign}(\mathsf{simsk},\cdot,\cdot),\mathsf{Sign}(\mathsf{simsk},\cdot)}(\mathsf{simpk})$
3 : assert $(\sigma \neq \sigma') \wedge \mathsf{Vrfy}(\mathsf{simpk}, m, \sigma) \wedge \mathsf{Vrfy}(\mathsf{simpk}, m, \sigma')$
4 : assert $\mathsf{pVrfy}(\mathsf{simpk}, m, Y, \widetilde{\sigma})$
5 : $y \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma); y' \leftarrow \mathsf{Extract}(Y, \widetilde{\sigma}, \sigma')$
6 : return $\mathcal{T}.\mathsf{Break}(\mathsf{simsk}, \mathcal{Q}_{\mathsf{Sign}} \cup \mathcal{Q}_{\mathsf{pSign}}, (m^*, \sigma^*), \mathsf{inst})$

$\mathsf{pSign}(\mathsf{simsk}, m, Y)$
1 : $\widetilde{\sigma} \leftarrow \mathcal{T}.\mathsf{SimPSign}(\mathsf{simsk}, m, Y)$
2 : $\mathcal{Q}_{\mathsf{pSign}}[m] \xleftarrow{\cup} \{Y, \widetilde{\sigma}\}$
3 : return $\widetilde{\sigma}$

$\mathsf{Sign}(\mathsf{simsk}, m)$
1 : $\sigma \leftarrow \mathcal{T}.\mathsf{SimSign}(\mathsf{simsk}, m)$
2 : $\mathcal{Q}_{\mathsf{Sign}}[m] \xleftarrow{\cup} \{\sigma\}$
3 : return $\sigma$
```

Figure 19: The second game $\mathcal{G}_2(\lambda)$.

Lemma 3 (Unlinkability). *Let* $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ *be a homomorphic one-way function, and* Rel *be a canonical hard relation with auxiliary input for* OWF*. Let* Σ *be a dichotomic signature scheme with respect to* OWF *and let* AS *be a dichotomic adaptor signature scheme for both* Σ *and* Rel *as per Construction 1. Then,* AS *is unlinkable as for Definition 14.*

Proof of Lemma 3. Lemma 3 follows directly by Lemma 22. □

Lemma 4 (Pre-Verify Soundness). *Let* $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ *be a homomorphic one-way function, and* Rel *be a canonical hard relation with auxiliary input for* OWF*. If the algorithm* $\mathsf{StmtVrfy}$ *can check the membership of the statement in the language* $\mathcal{L}_{\mathsf{Rel}}$ *efficiently, Construction 1 achieves statistical pre-verify soundness (c.f. Definition 16).*

Proof of Lemma 4. If the algorithm $\mathsf{StmtVrfy}$ checks the language membership of a statement Y efficiently, then the pre-verification algorithm of Construction 1 returns 0 for all statements which are not in the language of the relation. Therefore, Construction 1 achieves statistical pre-verify soundness. □

8.2 Adaptor Signatures from BBS^+

This section shows that the BBS^+ signature scheme [BBS04; ASM06; CDL16b] achieves all conditions required for Theorem 4. The BBS^+ signature scheme is pairing-based and provably secure in the standard model, assuming the hardness of the $q-$SDH assumption [BB04]. Below, we define the $q-$SDH assumption and the BBS^+ signature scheme.

Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_t$ be three cyclic groups of prime order p. Let g_0 be a generator of $\mathbb{G}_1$ and h_0 be a generator of $\mathbb{G}_2$, such that $g_0 = \psi(h_0)$ for an efficient isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ and let $\widetilde{e}$ be an efficiently computable bilinear map $\widetilde{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$.

Definition 25 (q-Strong Diffie-Hellman Problem). *The $q-$Strong Diffie-Hellman (q-SDH) problem in* $(\mathbb{G}_1, \mathbb{G}_2)$ *is defined as follows: On input a $(q+2)$-tuple* $(g_0, h_0, h_0^x, h_0^{(x^2)}, \cdots,$

$h_0^{(x^q)}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$, *where as above* $g_0 = \psi(h_0)$, *output a pair* $(c, g_0^{\frac{1}{x+c}})$ *where* $c \in \mathbb{Z}_p^*$. *An algorithm* $\mathcal{A}$ *has advantage* ϵ *in solving the q-SDH problem in* $(\mathbb{G}_1, \mathbb{G}_2)$ *if*

$$\Pr\left[\mathcal{A}(g_0, h_0, h_0^x, h_0^{(x^2)}, \cdots, h_0^{(x^q)}) = (c, g_0^{\frac{1}{x+c}})\right] \geq \epsilon$$

where the probability is taken over the random choice of the generator $h_0 \in \mathbb{G}_2$ *with* $g_0 = \psi(h_0)$, *the random choice of* $x \in \mathbb{Z}_p^*$, *and the random bits consumed by* $\mathcal{A}$.

Based on the strong Diffie-Hellman problem, we now define the strong Diffie-Hellmann assumption.

Definition 26 (q-Strong Diffie-Hellman Assumption)**.** *We say that the* (q, t, ϵ)*-SDH assumption holds in* $(\mathbb{G}_1, \mathbb{G}_2)$ *if no t-time algorithm has advantage at least* ϵ *in solving the q-SDH problem in* $(\mathbb{G}_1, \mathbb{G}_2)$.

The work of Boneh and Boyen [BB04] proves a lower bound on the complexity of the q-SDH problem in a generic group according to Shoup [Sho97].

Definition 27 (BBS+ Signature Scheme)**.** *Let* $\lambda \in \mathbb{N}$ *be the security parameter, the* BBS^+ *signature scheme consists of the following algorithms:*
$\underline{\mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda)}$. *The public parameter generation algorithm chooses* $g_0, g_1, \cdots, g_{L+1} \in \mathbb{G}_1$ *which are all generators of* $\mathbb{G}_1$ *and* h_0 *which is a generator of* $\mathbb{G}_2$. *The elements* g_i *and* $h_i := \psi^{-1}(g_i)$ *are chosen so that the relative discrete logarithm of the generators is unknown.*
$\underline{\mathsf{KGen}(1^\lambda)}$. *The key generation algorithm randomly chooses* $\mathsf{sk} \in \mathbb{Z}_p^*$ *and computes* $\mathsf{vk} = h_0^{\mathsf{sk}}$. *The secret key is* sk, *and the public key is* vk.
$\underline{\mathsf{Sign}(\mathsf{sk}, m_0, \cdots, m_L)}$. *To sign a tuple* $(m_1, \cdots, m_L) \in \mathbb{Z}_p^*$ *of messages, the signing algorithm chooses a value* $e \in \mathbb{Z}_p^*$ *and a random number* r. *It then computes* $A = [g_0 \cdot g_1^r \cdot g_2^{m_1} \cdots g_{L+1}^{m_L}]^{\frac{1}{e+\mathsf{sk}}}$. *It outputs the signature* $\sigma := (A, e, r)$.
$\underline{\mathsf{Vrfy}(\mathsf{vk}, \sigma, m)}$. *To verify a signature* (A, e, r) *on a message tuple* $(m_1, \ldots, m_L)$, *the verification algorithm checks, if* $\widetilde{e}(A, \mathsf{vk} \cdot h_0^e) = \widetilde{e}(g_0 \cdot g_1^r \cdot g_2^{m_1} \cdots g_{L+1}^{m_L}, h_0)$.

Applying Theorem 4 to BBS^+**.** The following lemmas show that BBS^+ is a dichotomic signature scheme, and its strong unforgeability can be proven using a simulatable transparent reduction. Furthermore, we show the existence of an efficient statement verification algorithm if the group $\mathbb{G}_1$ has efficient membership checking.

Lemma 5. *The BBS+ signature scheme is a dichotomic signature scheme w.r.t the homomorphic function* $\mathsf{OWF} : y \in \mathbb{Z}_p \to g_1^y \in \mathbb{G}_1$ *in accordance with Definition 17.*

Proof of Lemma 5. Dichotomic signature schemes are defined with respect to a homomorphic one-way function OWF. The provided function OWF is homomorphic, since $g_1^{x+y} = g_1^x \cdot g_1^y$. We now show that BBS+ is a dichotomic signature scheme w.r.t. this homomorphic function. Therefore, we show decomposability, verifiability, and homomorphism separately. Decomposability follows through the definitions of the algorithms Σ_1 and Σ_2. For a signature $\sigma := (A, e, r)$, we set

$$\Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r)) := (A, e) = ([g_0 \cdot g_1^r \cdot g_2^{m_1} \cdots g_{L+1}^{m_L}]^{\frac{1}{e+\mathsf{sk}}}, e)$$

and $\Sigma_2(\mathsf{sk}, m, \sigma_1; r) := r$. The function Σ_1 can be computed on the inputs sk, $(m_1, \dots, m_L)$, $\mathsf{OWF}(r)$, and Σ_2 with the input r, so decomposability is achieved.Verifiability holds since the verification equation

$$\widetilde{e}(A, \mathsf{vk} \cdot h_0^e) = \widetilde{e}(g_0 \cdot g_1^r \cdot g_2^{m_1} \cdots g_{L+1}^{m_L}, h_0)$$

can be checked using the inputs A, e, g_1^r. The function Σ_2 is homomorphic, since $\Sigma_2(\mathsf{sk}, m, \sigma_1; r) := r$, and hence, $\Sigma_2(\mathsf{sk}, m, \sigma_1; r) + y = \Sigma_2(\mathsf{sk}, m, \sigma_1; r + y)$. □

Lemma 6. *If the group $\mathbb{G}_1$ has efficient membership testing, then there exists an efficient algorithm* $\mathsf{StmtVrfy}$ *that, on input* $(\mathsf{vk}, Y, \mathsf{aux})$*, can decide if a statement Y is an element of the image of* OWF.

Proof. The function OWF is injective and maps from $\mathbb{Z}_p$ to $\mathbb{G}_1$. Both of these groups have order p. Hence, OWF is an isomorphism between the groups $\mathbb{G}_1$ and $\mathbb{Z}_p$. Hence, each group element is a valid statement. Since the group $\mathbb{G}_1$ has efficient membership testing, the algorithm $\mathsf{StmtVrfy}$ can return whatever the membership testing algorithm output. If the algorithm $\mathsf{StmtVrfy}$ outputs 1, Y is an element of the image of OWF and hence is a valid statement. □

Lemma 7. *The BBS+ signature scheme is strongly unforgeable against an adaptively chosen message attack under the q-SDH assumption. This strong unforgeability can be proven using a transparent reduction.*

Proof of Lemma 7. The proof of existential unforgeability is already given in the work of [ASM06]. Without changing the proof, we add some arguments as to why this proof also shows strong unforgeability. Moreover, we match the main steps of the proof to the interfaces of a transparent reduction. For this proof, we assume there exists an adversary $\mathcal{A}$ against the strong unforgeability of BBS+ and a reduction $\mathcal{R}$ that uses $\mathcal{A}$ to solve the q-SDH problem. We, furthermore, assume the adversary $\mathcal{A}$ makes q signature queries. The reduction $\mathcal{R}$ has as input an instance of the q-SDH problem, namely $(g_0, h_0, h_0^x, h_0^{(x^2)}, \cdots, h_0^{(x^q)}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$. Applying the technique of Boneh and Boyen in the proof of Lemma 3.2 in [BB04], $\mathcal{R}$ obtains generators g_0, h_0 of $\mathbb{G}_1$ and $\mathbb{G}_2$ and a public key $\mathsf{vk} = h_0^x$. Furthermore, $\mathcal{R}$ gets $q-1$ SDH pairs (B_i, e_i), such that $\widetilde{e}(B_i, \mathsf{vk}h_0^{e_i}) = \widetilde{e}(g_0, h_0)$ for each i. Due to the technique of the same lemma, any new SDH pair (B, e) besides these $q-1$ pairs leads to the solution of the original $q - SDH$ problem. We now show how the reduction answers the q signature queries. $\mathcal{R}$ randomly selects $e^*, a^*, k^* \leftarrow\$ \mathbb{Z}_p^*$ and computes $h_1 = [(\mathsf{vk}\cdot h_0^{e^*})^{k^*} \cdot h_0^{-1}]^{1/a^*}$. Note that $h_1 = h_0^{\frac{(e^*+x)k^*-1}{a^*}}$. For $j = 2 \cdots L+1$, where L is the maximum length of the block of messages, randomly select $\mu_j \leftarrow\$ \mathbb{Z}_p^*$ and compute $h_j = h_1^{\mu_j}$. Furthermore, $\mathcal{R}$ samples a random number $1 \leq i^* \leq q$. This computations on input a instance of the q-SDH problem lead to the system parameters $(g_0, h_0, h_1, \cdots, h_{L+1})$, the simulated public key $\mathsf{simpk} = h_0^x$ and the simulated secret key $\mathsf{simsk} = (e^*, i^*, a^*, k^*, (B_1, e_1), \cdots, (B_{q-1}, e_{q-1}), \mu_2, \cdots, \mu_{L+2})$. We identify these computations with the transparent reduction's simulated key generation algorithm SimKg. We identify the i^*-th query as query $*$. We suppose the message block to be signed for the $i-$th query is $(m_{1,i}, \cdots, m_{l_i,i})$ such that $l_i \leq L$. For the $q-1$ signature queries other than query $*$, $\mathcal{R}$ answers by using the $q-1$ SDH pairs (B_i, e_i) as follows. It randomly selects $r_i \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and computes $a_i = r_i + t_i$, where $t_i = m_{1,i}\mu_2 + \cdots + m_{l_i,i}\mu_{l_i+1}$. We note, that due to the construction of $h_i = h_1^{\mu_i}$, $g_2^{m_{1,i}} \cdots g_{l_i+1}^{m_{l_i,i}} = g_1^{t_i}$. Afterwards, $\mathcal{R}$ computes $A_i = [g_0 g_1^{a_i}]^{\frac{1}{e_i+\mathsf{sk}}}$ and returns the signature for the i-th query as (A_i, e_i, r_i). Using the simulated secret key simsk, this value A_i can be computed without knowing the real secret key sk by evaluating

$$\begin{aligned}
A_i = [g_0 g_1^{a_i}]^{\frac{1}{e_i+\mathsf{sk}}} &= B_i g_1^{\frac{a_i}{\mathsf{sk}+e_i}} \\
&= B_i [g_0^{\frac{a_i k^*(e^*+\mathsf{sk})-a_i}{(\mathsf{sk}+e_i)a^*}}] \\
&= B_i^{(1-\frac{a_i}{a^*})}([g_0^{\frac{a_i k^*}{a^*}}]^{(1-\frac{e_i-e^*}{e_i+\mathsf{sk}})}) \\
&= (B_i^{(1-\frac{a_i}{a^*}-\frac{(e_i-e^*)a_i k^*}{a^*})})(g_0^{\frac{a_i k^*}{a^*}}).
\end{aligned}$$

For the query $*$, the reduction chooses r^*, such that $r^* + t_i = a^*$. Afterwards, it computes $A^* = g_0^{k^*}$ and returns (A^*, e^*, r^*) as signature. We identify these computations with the simulated signing algorithm of a transparent reduction that on input a message m and the simulated signing key simsk outputs simulated signatures. Finally, the adversary $\mathcal{A}$ outputs a forged signature $\sigma' := (A', e', r')$ on a message $(m'_1, \cdots, m'_{l'})$. We now show how to use this forgery to break the instance of the q-SDH problem. This description matches the breaking algorithm of a transparent reduction. For the forgery σ', there are three possibilities:

- The value e' is distinct from all previous values e that were output by the reduction.
- The value e' corresponds to some previous value e_i and the value A' matches the value A_i.
- The value e' corresponds to some previous value e_i and the value A' does not match the value A_i.

We show how to break the q-SDH problem for each case separately.

- Case I [$e' \notin e_i, e^*$]: Denote $y = r' + m'_1\mu_2 + \cdots + m'_{l'}\mu_{l'+1}$. The reduction computes

  $$B' = g_0^{\frac{1}{e'+\mathsf{sk}}} = [A' g_0^{\frac{-k^* y}{a^*}}]^{\frac{a^*}{a^*-y-k^* y(e'-e^*}}$$

  and outputs (B', e'). Since $e' \notin e_i, e^*$, this is a new SDH pair.

- Case II [$e' = e_i$ and $A' = A_i$ or $e' = e^*$ and $A' = A^*$: This happens only if the adversary can compute the discrete logarithms of two of the h_is. This happens only with negligible probability. If $m' = m_i$ or $m' = m^*$, this is not a valid forgery, since then $(A', e', r') = (A_i, e_i, r_i)$, or $(A', e', r') = (A^*, e^*, r^*)$.

- Case III [$e' \in e_i, e^*$ and $A' \neq A_i$ or $A' \neq A^*$: With probability $\frac{1}{q}$, it holds, that $e' = e^*$. We define y as in Case I. The reduction outputs the new pair (B^*, e^*), where

  $$B^* = g_0^{\frac{1}{e^*+\mathsf{sk}}} = [A' g_0^{\frac{-k^* y}{a^*}}]^{\frac{a^*}{a^*-y}}.$$

  This is a new SDH tuple, as the value e^* was never part of an input-SDH tuple.

This breaking algorithm has success probability of ϵ/q, where ϵ is the success probability of the adversary $\mathcal{A}$. Therefore, the reduction $\mathcal{R}$ can break the q-SDH assumption with non-negligible probability whenever an efficient adversary against the BBS+ signature scheme exists that has a non-negligible advantage. $\square$

Lemma 8. *Let g_1 be a generator of $\mathbb{G}_1$, and let the values $C_1, \ldots, C_Q \in \mathbb{G}_1$ be randomly distributed. Let* Rel *be a hard relation with publicly decideable auxiliary information w.r.t the language $\mathcal{L}_{\mathsf{Rel}} := \{(Y, \mathsf{aux}, y) | y \in \mathbb{Z}_p^*; Y := g_1^y; aux := (C_1^y, \ldots, C_Q^y)\}$. The transparent reduction of the SUF-CMA security of the BBS+ signature scheme is simulatable for* Rel *in accordance with Definition 23.*

Proof of Lemma 8. We update the simulated key generation algorithm of the transparent reduction to output the public parameters of the hard relation. These are $(g_1, C_1, \ldots, C_Q)$, where g_1 is the same element used for the signature scheme. The values $C_1, \ldots C_Q$ are computed the following way: The reduction samples Q random values $\zeta_1, \ldots \zeta_Q \leftarrow\!\!\$\ \mathbb{Z}_p^*$ and computes $C_i = B_i^{\zeta_i}$ for $1 \leq i \leq q$ and $i \neq i^*$. All other values C_j are computed via $g_0^{\zeta_j}$. All the values C_k are distributed randomly. and hence have the same distribution as in an honest execution. Therefore, with this modification, we still have a transparent reduction. Now we have to provide the algorithm Sim: For the pre-signature query on input a message m, a statement Y, and the auxiliary information aux, the reduction $\mathcal{R}$ checks the auxiliary information using $\mathsf{AuxVrfy}$. The aux value can be checked by verifying the equation $\widetilde{e}(Y, \psi^{-1}(C_i) = \widetilde{e}(g_1, \psi^{-1}(\mathsf{aux}_i))$. Afterward, the reduction runs the simulated signing algorithm $\mathsf{SimSign}$, such that the current signing index $i \neq i^*$ to obtain a simulated signature $\sigma_i = (A_i, e_i, r_i)$. This can be done by skipping i^* for pre-signatures but using it for normal signatures afterward. The reduction modifies this value A_i to $A'_i := A_i \cdot g_1^{\frac{y}{e_i + \mathsf{sk}}} = [g_0 \cdot g_1^{r+y} \cdot g_2^{m_1} \cdots g_{L+1}^{m_L}]^{\frac{1}{e_i+\mathsf{sk}}}$. This computation is possible since the $\mathcal{R}$ knows the auxiliary value $\mathsf{aux}_i = C_i^y = B_i^{\zeta_i \cdot y} = g_0^{\frac{\zeta_i \cdot y}{e_i + \mathsf{sk}}}$. Using the identity $h_1 = h_0^{\frac{(e^*+x)k^*-1}{a^*}}$, $\mathcal{R}$ can compute

$$C_i^{\frac{1}{\zeta_i} \cdot \frac{(e^*+x)k^*-1}{a^*}} = g_0^{\frac{\zeta_i \cdot y}{e_i+\mathsf{sk}} \cdot \frac{1}{\zeta_i} \cdot \frac{(e^*+x)k^*-1}{a^*}} = g_1^{\frac{\zeta_i \cdot y}{e_i+\mathsf{sk}} \cdot \frac{1}{\zeta_i}} = g_1^{\frac{y}{e_i+\mathsf{sk}}}.$$

By the construction, the values (A'_i, e_i, r_i) are a valid pre-signature for the message m and the statement Y. Furthermore, this pre-signature is identically distributed to non-simulated pre-signatures, and hence, the transparent reduction achieves simulatability using the algorithm Sim. $\square$

8.3 Adaptor Signatures from Partitioned Signatures

The work of Boneh, Shen, and Waters [BSW06] provides a compiler, which we refer to as *BSW*, that transforms partitioned signatures that achieve existential unforgeability into dichotomic signatures that achieve strong unforgeability. We define partitioned signatures in this section and show that the output signatures of the *BSW* compiler have a simulatable transparent reduction. A famous partitioned signature is the Waters signature [Wat05]. Using the *BSW* compiler, we obtain Waters^+: a strong, unforgeable version of the Waters signature scheme. As a result of Section 8.1, Waters^+ can be transformed into an adaptor signature scheme using Construction 1.

Definition 28 (Partitioned Signatures [BSW06]). *We say that a signature scheme* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *is* partitioned, *if it satisfies two properties:*
Decomposition: The signing algorithm Sign *can be broken into two deterministic algorithms* F_1 *and* F_2 *so that a signature* $\sigma = (\sigma_1, \sigma_2)$ *on a message* m *using the singing key* sk *is computed as:*

$$r \leftarrow\!\!\$\ \mathcal{D}_{\mathsf{R}}; \sigma \leftarrow (F_1(sk, m; r), F_2(\mathsf{sk}; r))$$

Injectivity: Given a message m *and a* σ_2*, there is at most one* σ_1 *so that* (σ_1, σ_2) *verifies as a valid signature, i.e.* $\mathsf{Vrfy}(\mathsf{vk}, m, (\sigma_1, \sigma_2) = 1$.

Based on partitioned signatures, Boneh, Shen, and Waters provide a compiler to obtain new signature schemes.

Construction 2 (The *BSW* Signature Compiler). *Let* $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *be a partitioned signature scheme where the signing algorithm can be decomposed into the two functions* F_1 *and* F_2*. Let* $\mathbb{G}$ *be a group of prime order* p *and let* $\mathcal{H} = \{\mathcal{H}(k, \cdot)\}$ *be collision-resistant hash function family that maps elements into the group* $\mathbb{Z}_p$*. The Boneh-Shen-Waters transformation transforms this signature scheme* Σ *into a novel signature scheme* Σ' *as depicted in Fig. 20.*

```
$\mathsf{KGen}(\lambda)$
1 : $g \leftarrow\!\!\$\ \mathbb{G}$    // generator of $\mathbb{G}$
2 : $h \leftarrow\!\!\$\ \mathbb{G}$    // generator of $\mathbb{G}$
3 : $k \leftarrow\!\!\$\ \mathcal{D}_{\mathsf{K}}$    // a key for $\mathcal{H}$
4 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\lambda)$
5 : $\mathsf{vk}' \leftarrow (\mathsf{vk}, g, h, k)$
6 : **return** $(\mathsf{vk}', \mathsf{sk})$
```

```
$\mathsf{Sign}'(\mathsf{sk}', m; s)$
1 : $r \leftarrow\!\!\$\ \mathcal{D}_{\mathsf{R}}$
2 : $\sigma_2 \leftarrow F_2(\mathsf{sk}; r)$
3 : $t \leftarrow \mathcal{H}(k, m||\sigma_2) \in \mathbb{Z}_p$
4 : $m' \leftarrow g^t h^s \in \mathbb{G}$
5 : $\sigma_1 \leftarrow F_1(\mathsf{sk}, m'; r)$
6 : **return** $(\sigma_1, \sigma_2, s)$
```

```
$\mathsf{Vrfy}'(\mathsf{vk}', m, \sigma)$
1 : $(\sigma_1, \sigma_2, \sigma_3) \leftarrow \sigma$
2 : $\widetilde{t} \leftarrow \mathcal{H}(k, m||\sigma_2) \in \mathbb{Z}_p$
3 : $\widetilde{m} \leftarrow g^{\widetilde{t}} h^{\sigma_3}$
4 : **return** $\mathsf{Vrfy}\,(\mathsf{vk}, \widetilde{m}, (\sigma_1, \sigma_2))$
```

Figure 20: The BSW transformation [BSW06].

Partitioned signatures are not necessarily dichotomic since, in general, it is impossible to compute F_1 using as input $\mathsf{OWF}(r)$ instead of r. Furthermore, the CL^+ signature scheme is dichotomic (c.f. Lemma 13), but is not a partitioned signature, as for a fixed $\sigma_2 := r$, there exist multiple values v, e such that $v^e \equiv a^m \cdot b^r \cdot c \bmod n$. However, when we apply the *BSW* compiler to partitioned signatures, the output signature is indeed dichotomic, as we show in Lemma 9.

Applying Theorem 4 to ***BSW*****.** We now show that the output of the *BSW* compiler yields a strongly unforgeable dichotomic signature with a simulatable transparent reduction.

Lemma 9. *The BSW signature compiler (c.f. Construction 2) yields a dichotomic signature scheme w.r.t. the homomorphic function* $\mathsf{OWF} : y \in \mathcal{D}_{\mathsf{R}} \to h^y \in \mathbb{G}$*, where* $h \in \mathsf{vk}'$*.*

Proof of Lemma 9. The function OWF is a DLog function and hence homomorphic. Since we can assume that $|\mathcal{D}_\mathsf{R}| = |\mathbb{G}|$, and h is a generator of $\mathbb{G}$, this function is also injective. We now show that BSW is a dichotomic signature scheme w.r.t. OWF. Therefore, we show decomposability, verifiability, and homomorphism separately. Note that the BSW construction uses two random elements, namely r and s, and we use s to show decomposability. Decomposability follows through the definitions of the algorithms Σ_1 and Σ_2: For a signature $\sigma := (\sigma', \sigma'', s)$, we set

$$\sigma_1 := \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(s)) = (\sigma', \sigma'') = \left(F_1(\mathsf{sk}, g^{\mathcal{H}(k, m||F_2(\mathsf{sk},r))} \cdot \mathsf{OWF}(s); r), F_2(\mathsf{sk}; r)\right),$$

and $\Sigma_2(\mathsf{sk}, m, \sigma_1; s) := s$. The function Σ_1 can be computed on the inputs $\mathsf{sk}, m; \mathsf{OWF}(s)$, and Σ_2 using the input s, so decomposability is achieved. Verifiability holds since the verify equation parses σ into $\sigma_1 = (\sigma', \sigma'')$ and $\sigma_2 = s$. It afterwards only uses $h^s = \mathsf{OWF}(\sigma_2)$ to verify the signature. BSW is homomorphic, since $\sigma_2 + y = s + y = \Sigma_2(\mathsf{sk}, m; s+y)$. □

Lemma 10. *If the group $\mathbb{G}$ has efficient membership testing, then there exists an efficient algorithm* $\mathsf{StmtVrfy}$ *that, on input* $(\mathsf{vk}, Y, \mathsf{aux})$*, can decide if a statement Y is an element of the image of* OWF.

Proof. This proof follows the argument of the proof of Lemma 14. □

Proposition 4 (Theorem 1 from [BSW06])**.** *Let Σ be a signature scheme that is $(t, q, \epsilon/3$-existentially unforgeable, let $\mathbb{G}$ be a group in which the $(t, \epsilon/3)$-DLog assumption holds and let $\mathcal{H}$ be $(t, \epsilon/3)$-collision-resistent. Then the signature scheme Σ' that is built by Construction 2 is (t, q, ϵ)-strongly existentially unforgeable (SUF-CMA secure).*

Lemma 11. *The strong unforgeability of Proposition 4 can be proven using a transparent reduction.*

Proof of Lemma 11. In this proof, we recite the main part of the proof of Theorem 1 in [BSW06]. This means we elaborate on how the reduction behaves if a so-called Type III forger tries to break strong unforgeability. We do not consider the other cases in this proof, since the reductions against the Type II forger and the Type I forger reduce the hardness of the strong unforgeability to the collision-resistance of the hash function $\mathcal{H}$ and the hardness of the DLog assumption. Therefore, these reductions have access to the signing key sk', and hence, these reductions are trivially transparent. We refer the reader to [BSW06] for a full proof of strong unforgeability.

The reduction for a Type III forger has an underlying hardness assumption of the existential unforgeability of the input signature scheme. This is an interactive hardness assumption. For the proof, we assume there exists a forger $\mathcal{A}$ that can ask for signatures on the messages $m_1, \ldots, m_q$ and learns signatures $\sigma_i = (\sigma_i', \sigma_i'', s_i)$ on these messages from the reduction. Let $t_i = \mathcal{H}(k, m_i||\sigma_i'')$, and $m_i' = g^{t_i} h^{s_i}$ for $1 \leq i \leq q$. Let $((m^*, \sigma^* = (\sigma'^*, \sigma''^*, s^*))$ be the forgery produced by $\mathcal{A}$. Let $t^* = \mathcal{H}(k, m||\sigma''^*)$ and let $m'^* = g^{t^*} h^{s^*}$. A Type III forger is a forger for which $m'^* \neq m_i' \; \forall 1 \leq i \leq q$.

Suppose $\mathcal{A}$ is a Type III forger. We construct a simulator $\mathcal{B}$ that breaks the existential unforgeability of Σ. $\mathcal{B}$ receives as input a public key vk of Σ. $\mathcal{B}$ runs $\mathcal{A}$ as follows: To compute the simulated public key simpk, $\mathcal{B}$ samples a random generator $g \in \mathbb{G}$ and a random number $a \in \mathbb{Z}_p^*$. It sets $h := g^a$. Then it selects a hash key $k \in \mathcal{D}_\mathsf{K}$ uniformly at random and provides the public key $\mathsf{vk}' := (\mathsf{vk}, g, h, k)$ to $\mathcal{A}$. We identify this procedure as *simulated key generation* SimKg with a simulated public key $\mathsf{simpk} = \mathsf{vk}'$ and simulated secret key $\mathsf{simsk} = a$. To answer signature queries, on a message m, the simulator $\mathcal{B}$ samples $\omega \leftarrow\!\$\ \mathbb{Z}_p$ uniformly at random and sets $m' = g^\omega$. Then $\mathcal{B}$ queries its own signature oracle for a signature on the message m' and learns (σ', σ''). To modify this signature to a valid signature for Σ', $\mathcal{B}$ computes $t \leftarrow \mathcal{H}(k, m||\sigma_2)$ and $s = (\omega - t)/a$. It then returns (σ_1, σ_2, s) to $\mathcal{A}$. This is a valid signature on the message m, since $m' = g^\omega = g^{as+t} = g^t h^s$ and s is uniform in $\mathbb{Z}_p$. We refer to this signing protocol as the *simulated signing algorithm* SimSign. When the forger $\mathcal{A}$ outputs its forgery $(m^*, (\sigma'^*, \sigma''^*, s^*))$, $\mathcal{B}$ outputs $(m'^*, (\sigma'^*, \sigma''^*))$ and breaks the EUF-CMA security of Σ, which is the underlying hardness assumption. This is the Break algorithm; hence, this reduction is transparent. □

Lemma 12. *The transparent reduction of the SUF-CMA security of the BBS+ signature scheme is simulatable (c.f. Definition 23) for the relation* $\mathsf{OWF} : y \in \mathcal{D}_\mathsf{R} \to h^y \in \mathbb{G}$ *with empty auxiliary information. The element h is part of the signer's public key (c.f. Construction 2).*

Proof of Lemma 12. To prove simulatability, we have to show how the simulator $\mathcal{B}$ in the proof of Lemma 11 can successfully simulate pre-sign queries on input a message m and a statement $Y = h^y$. $\mathcal{B}$ does this the following way: It first samples a random exponent ω and computes $m' = g^\omega \cdot Y$. Then it proceeds like in the SimSign algorithm by querying a signature on m' and computing $t \leftarrow \mathcal{H}(k, m||\sigma_2)$ and $s = (\omega - t)/a$. This is a valid pre-signature, since $m' = g^\omega \cdot Y = g^{as+t+ay} = g^t h^s h^y$. □

Waters$^+$ **Adaptor Signatures.** The work of Boneh, Shen, and Waters [BSW06] shows that the Waters signature scheme is existentially unforgeable and partitioned. Hence, we can obtain Waters$^+$ by applying the Waters signature to the *BSW* compiler. The Waters signature scheme is based on the CDH assumption without using random oracles. This leads us to the first secure adaptor signature scheme based on the CDH assumption. We show the structure of the Waters signature scheme in Fig. 21 and refer the reader to the work of Boneh, Shen, and Waters [BSW06] for formal proofs of partition and existential unforgeability.

8.4 Adaptor Sigantures from CL$^+$

Our third example of a dichotomic adaptor signature scheme in the standard model is the Camenisch-Lysyanskaya (CL) signature scheme [CL03] that is provably secure assuming the hardness of the strong RSA assumption. We define a slightly modified version of the CL scheme that we call CL$^+$. We have to modify the CL signature scheme since our security analysis (c.f. Section 8.1) relies on the SUF-CMA security of the underlying signature scheme. Still, the definition of the CL signature scheme, as provided in [CL03] achieves only EUF-CMA security. CL$^+$ has slightly tuned parameters compared to the original CL scheme. In addition, we define the CL$^+$ scheme over the signed quadratic residues QR_n^+ [HK09] instead of the quadratic residues QR_n. The signed quadratic residues behave similarly compared to the quadratic residues but allow for efficient group membership checking. Looking ahead, this will guarantee pre-verify soundness for the CL$^+$ adaptor

```
$\mathsf{KGen}(\lambda)$
1 : $g \leftarrow\!\$\ \mathbb{G}$    // a generator of $\mathbb{G}$
2 : $\alpha \leftarrow\!\$\ \mathbb{Z}_p$
3 : $g_1 := g^{\alpha}$
4 : $g_2, u', u_1, \ldots, u_n \leftarrow\!\$\ \mathbb{G}$
5 : $U := (u_1, \ldots, u_n)$
6 : $\mathsf{vk} \leftarrow (g, g_1, g_2, u', U)$
7 : $\mathsf{sk} \leftarrow g_2^{\alpha}$
8 : return $(\mathsf{vk}, \mathsf{sk})$
```

```
$\mathsf{Sign}(\mathsf{sk}, m)$
1 : $\sigma_1 \leftarrow \mathsf{sk} \cdot \left(u' \prod_{i=1}^{n} u_i^{m_i}\right)^r \in \mathbb{G}$
2 : $\sigma_2 \leftarrow g^r \in \mathbb{G}$
3 : return $(\sigma_1, \sigma_2)$
```

```
$\mathsf{Vrfy}(\mathsf{vk}, m, \sigma)$
1 : $(\sigma_1, \sigma_2) \leftarrow \sigma$
2 : $b = e(\sigma_1, g) \overset{?}{=} e\left(\sigma_2, u' \prod_{i=1}^{n} u_i^{m_i}\right) \cdot e(g_1, g_2)$
3 : return $b$
```

Figure 21: The Waters signature scheme [Wat05; BSW06].

signatures. By $QR_n^+ \subseteq \mathbb{Z}_n^*$ we denote the set of quadratic residues modulo n and postpone the remaining number-theoretic basics of CL^+ to Appendix B.1.

Definition 29 (Modified Camenisch Lysyanskaya Signatures CL^+)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, $\ell_m, \ell_n, \ell_r \in \mathbb{N}$. The Camenisch Lysyanskaya signature scheme for the message space $\mathcal{D}_{\mathsf{M}} = \{0,1\}^{\ell_m}$ consists of the following algorithms:*
$\underline{\mathsf{pp} \leftarrow \mathsf{Pgen}(1^\lambda)}$. *The public parameter generation algorithm chooses a special RSA modulus $n = pq$, such that q and q are safe primes, i.e., $p = 2p' + 1, q = 2q' + 1$ for primes p' and q', and n is a Blum integer, i.e. $n \equiv 3 \mod 4$. In addition, the primes p and q have the lengths $\ell_n = 2\lambda$. In addition, the key generation algorithm samples a generator u of QR_n^+.*
$\underline{(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)}$. *The key generation algorithm chooses $a, b, c \in QR_n^+$ uniformly at random such that there exists an $s \in \mathbb{Z}_n^*$, such that $u^s \equiv b$ mod n. It outputs the public key $\mathsf{vk} := (n, a, b, c)$ and the secret key $\mathsf{sk} := (p, s)$.*
$\underline{\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)}$. *The signing algorithm chooses a prime number e of length $\ell_e \geq \ell_r + 3$, and a random number r of length $\ell_r = \ell_n + \ell_m + l$ where l is a security parameter. Finally, it computes a v such that $v^e \equiv a^m b^r c \mod n$. It returns the triple $\sigma := (e, v, r)$.*
$\underline{b \leftarrow \mathsf{Vrfy}(\mathsf{vk}, m, \sigma)}$. *The verification algorithm verifies that the triple (e, v, r) is a signature $m \in \mathcal{D}_{\mathsf{M}}$ by checking that $v^e \equiv a^m b^r c$ mod n and checking that $2^{\ell_e} > e > 2^{\ell_e - 1}$ and that $r < 2^{\ell_r}$.*

Modifications of CL^+. The definition of CL^+ has modifications compared to the original CL signature scheme. Firstly, the size of ℓ_e is increased from $\ell_e \geq l_m + 2$ to $\ell_e \geq l_r + 3 = l_n + l_m + l + 3$. This modification gives the strong unforgeability of CL^+ signatures, and signatures generated with this adaption are also valid in the original scheme as $l_r + 2 > l_m + 2$. Secondly, the secret key has an additional element s, which is relevant when considering the adaptor construction but does not change the signing or the verification process. The generator u is part of the public parameters. Thirdly, the verification algorithm checks that $r < 2^{\ell_r}$ to ensure that the random element is in the right domain. Finally, we define CL^+ over the group of the *signed* quadratic residues modulo n. This allows efficient membership checking of the group elements while not influencing the strong RSA assumption (c.f. Theorem 7).

Applying Theorem 4 to CL^+. In this section, we show that the CL^+ signature scheme is dichotomic, and its strong unforgeability can be proven using a simulatable transparent reduction. In addition, we show the existence of an efficient statement verification algorithm.

Lemma 13. *The CL^+ signature scheme (Definition 29) is a dichotomic signature scheme w.r.t the homomorphic function $\mathsf{OWF} : y \in \mathbb{Z}_n^* \to b^y \in QR_n^+$ in accordance with Definition 17.*

Proof. The provided function OWF is homomorphic and quasi-injective. It is quasi-injective since if one finds two elements $x \neq y$, such that $b^x = b^y$, then $x - y$ divides the group order of QR_n^+, which allows factoring n. To show that the CL^+ signature is dichotomic, we must show decomposability, verifiability, and homomorphism. For decomposability, we define Σ_1 as $\Sigma_1(\mathsf{sk}, m, \mathsf{OWF}(r)) := (v, e)$, where $v^e \equiv a^m \cdot \mathsf{OWF}(r) \cdot c \bmod n$ and $\Sigma_2(\mathsf{sk}, m, \sigma_1; r) := r$. Verifiability holds as the verification equation $v^e \equiv a^m b^r c \bmod n$ can be checked using the inputs $\mathsf{vk}, m, v, e, \mathsf{OWF}(r) = b^r$. With the same argument as in the proof of Lemma 5, homomorphism holds. □

Lemma 14. *There exists an efficient algorithm* $\mathsf{StmtVrfy}$ *that, on input* $(\mathsf{vk}, Y, \mathsf{aux})$*, can decide if a statement Y is an element of the image of* OWF.

Proof. The function OWF maps surjective in the group QR_n^+, since it is quasi-injective, and $|QR_n^+| = \psi(n)/4 < \psi(n)$. Hence, each group element is a valid statement. Since the group QR_n^+ has efficient membership testing, the algorithm $\mathsf{StmtVrfy}$ can return whatever the membership testing algorithm output. If the algorithm $\mathsf{StmtVrfy}$ outputs 1, Y is an element of the image of OWF and hence is a valid statement. □

Lemma 15. *The CL^+ signature scheme is strongly unforgeable against an adaptively chosen message attack under the strong RSA assumption. This strong unforgeability can be proven using a transparent reduction.*

Proof of Lemma 15. The proof for the strong unforgeability of CL^+ is given in Appendix B. It remains to show that this proof uses a transparent reduction, which we do now. The strong RSA assumption on a flexible RSA instance is a non-interactive hard problem. We have to show that the reduction provided in the proof in Appendix B is, in fact, transparent. *Simulated Key Generation* is defined in the key generation algorithms for the odd and even case. The simpk equals the generated public key. And the simsk gets the values $e_1, \dots e_{2K}$, as well as $r_1, r_2, \alpha, \beta, t$, and u and s. The *Simulated Signature Generation* is also defined in the proof in Appendix B and all the used values to simulate the signatures are stored in simsk. The *Breaking Algorithm* checks if the forgery type was guessed right and calls the breaking algorithm of the right kind. □

Lemma 16. *Let (n, u) be an instance of the flexible RSA problem in the group QR_n^+ and $s \leftarrow\!\!\$\ \mathbb{Z} \setminus \{0\}$ a value. If DLog is hard in QR_n^+ with the generator u, then the function*

$\mathsf{OWF}_{u,s,n} : \mathbb{Z}_n^* \to QR_n^+$ *that maps* $x \in \mathbb{Z}_n^*$ *to* u^{sx} *is a quasi-injective homomorphic one-way function and thus builds a canonical hard relation* Rel. *This hard relation is a hard relation with privately decidable auxiliary information if* $\mathsf{AuxGen}(y) = u^y$ *for any witness* $y \in \mathbb{Z}_n^*$. *The transparent reduction of the SUF-CMA security of the* CL^+ *signature scheme is simulatable w.r.t* Rel *in accordance with Definition 23.*

Proof of Lemma 16. If DLog is hard in QR_n^*, and u is a generator of QR_n^*, then the value $\mathsf{aux} := u^y$ leads indeed to a hard relation with auxiliary information. As the signing key contains the value s, such that $u^s = b$ to the element $b \in \mathsf{vk}$, the pre-signer can verify this auxiliary information privately. The simulation Sim works the following way: Upon inputting a message m_i and a statement Y, it first checks if the auxiliary information is well-formed using s. Afterward, it selects the prime element e_i from simsk and samples a random $r \in \mathcal{D}_{\mathsf{R}}$. Using the auxiliary information $\mathsf{aux} = u^y$ from the statement it then computes $\mathsf{aux}' := \mathsf{aux}^{E/e}$ if the reduction is in the even state and else $\mathsf{aux}' := \mathsf{aux}^{E/e \cdot r_1}$. It then calculates v as in the signing algorithm of the transparent reduction and updates it to get $v' := v \cdot \mathsf{aux}' = v \cdot b^{y/e}$. This works by definition of the simulated key generation algorithm and returns (e, v', r). This leads to a valid pre-signature for the public key simpk on the message, statement pair (m, Y), as $v'^e \equiv a^m b^r Y c \bmod n$. Sim achieves the same distribution as the original pre-signing algorithm. Thus, no distinguisher can have a non-negligible advantage in distinguishing between a "normal" pre-signature and the output of Sim. □

8.5 Adaptor Signatures from ID-Based Sigantures

A recent work by Erwig et al. [Erw+21] showed that so-called commitment-recoverable identification schemes with a homomorphic transformation can be transformed into adaptor signatures. We slightly deviate from their formalization to match our generalized interface without sacrificing the compatibility to all known schemes, e.g., Schnorr [Sch91], the Katz-Wang [KW03], and Guillou-Quisquater [GQ90]. The following definitions are taken almost verbatim from [Erw+21].

Definition 30 (Canonical Identification Scheme). *A canonical identification scheme* ID *is defined as a tuple of four algorithms* ID := (IGen, P, ChSet, C) *defined as follows.*
<u>(sk, vk) ← IGen(λ).</u> *The key generation algorithm* IGen *takes as input the security parameter* λ *and outputs a secret and a public key* (sk, vk). *We assume that* vk *defines the set of challenges, namely* ChSet.
<u>P.</u> *The prover algorithm* P *consists of two algorithms namely* P_1 *and* P_2.

1. P_1 *takes as input the secret key* sk *and returns a commitment* $R \in \mathcal{D}_{\mathsf{rand}}$ *and a state* St.

2. P_2 *takes as input the secret key* sk, *a commitment* $R \in \mathcal{D}_{\mathsf{rand}}$, *a challenge* $h \in \mathsf{ChSet}$, *and a state* St *and returns a response* $s \in \mathcal{D}_{\mathsf{resp}}$.

<u>b ← V.</u> *The verifier algorithm* V *is a deterministic algorithm that takes the public key* vk *and the conversation transcript as input and outputs 1 (acceptance) or 0 (rejection).*

Definition 31. *An identification scheme* ID *is called* commitment-recoverable*, if* V *first internally calls a function* V_0 *which recomputes* $R_0 = \mathsf{V}_0(\mathsf{vk}, h, s)$ *and then outputs 1, iff.* $R_0 = R$.

Definition 32. *A signature scheme obtained from a canonical, commitment-recoverable identification scheme via the Fiat-Shamir heuristic has the following form.*

```
KGen(λ)
  1: $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{IGen}(\lambda)$
  2: return $(\mathsf{sk}, \mathsf{vk})$

Vrfy(vk, m, (h, s))
  1: $R := \mathsf{V}_0(\mathsf{vk}, h, s)$
  2: return $h = \mathcal{H}(R, m)$

Sign(sk, m)
  1: $(R, St) \leftarrow \mathsf{P}_1(\mathsf{sk})$
  2: $h := \mathcal{H}(R, m)$
  3: $s \leftarrow \mathsf{P}_2(\mathsf{sk}, R, h, St)$
  4: return $(h, s)$
```

Our formalization of dichotomic signatures captures all three signature schemes in [Erw+21], namely the signature schemes of Schnorr, Katz-Wang, and Guillou-Quisquater.

Lemma 17. *Signature schemes based on canonical, commitment-recoverable identification schemes as for Definition 32 are dichotomic signature schemes w.r.t. a one-way function* OWF *in accordance with Definition 17, if* P_2 *is homomorphic in the* St *component, the equation* $R = \mathsf{OWF}(St)$ *holds for all* St*, and there exists an algorithm* V_1*, such that* $V_0(\mathsf{vk}, h, s) = V_1(\mathsf{vk}, h, \mathsf{OWF}(s))$.

Proof. To show that these identification-scheme-based signature schemes are dichotomic, we have to provide definitions for the functions Σ_1, Σ_2, and Vrfy'. Note that we have, by assumption, that $\mathsf{OWF}(St) = R$. Our notation for adaptor signature schemes carries over to St being r and R being $\mathsf{OWF}(r)$. We identify the function Σ_1 to be $\mathcal{H}$ and Σ_2 to be P_2. These functions match the interfaces of Σ_1 and Σ_2. By the assumption of this theorem, Σ_2 is homomorphic in the randomness. As the identification-scheme is commitment-recoverable, we can define Vrfy' to be the check $h = \mathcal{H}(V_1(\mathsf{vk}, h, \mathsf{OWF}(s)), m)$. By the assumption on V_1, this fulfills the definition of Vrfy'. $\square$

In contrast to all previous schemes, the class of commitment-recoverable identification-based signature schemes has no fixed one-way function. While the Schnorr signature has a DLog one-way function and hence also has efficient image checking for the one-way function (c.f. Lemma 14), this does not generally hold for the Katz-Wang and Guillou-Quisquater signature schemes. Moreover, the hard relation between the group members of the Katz-Wang signature scheme cannot be checked without violating DDH. Therefore, if the Katz-Wang signature scheme should satisfy pre-verify soundness, a non-interactive zero-knowledge proof of membership needs to be inserted into the auxiliary information of each statement. Since the Identification-scheme-based adaptor signatures already make use of the random oracle, such a proof can be given using a Sigma protocol using the Fiat-Shamir heuristic.

Theorem 5. *Identification-scheme-based signature schemes as for Lemma 17 provide a simulatable transparent reduction $\mathcal{T}$ from the* SUF-CMA *security to an underlying interactive hard problem w.r.t the function* OWF*.*

Proof of Theorem 5. As the identification-scheme-based signature schemes are in the random oracle model, simulatability can be achieved without using a simulated secret key having access to a signing oracle. The reduction from SUF-CMA security to an underlying interactive hard problem is shown in [KMP16]. We do not need to specify the simulated signing and key-generation algorithm from the reduction since we do not need to use SimKg. Therefore, we implicitly use the algorithms from the StrongSigForge game. This means, as SimKg, we just forward the provided public key from the StrongSigForge game. The breaking algorithm requires just a valid message, forgery pair $(m^*, \sigma^*$ that breaks the SUF-CMA security of Σ. The simulated signing algorithm SimSign calls the signing oracle provided by the StrongSigForge game and forwards the output. The simulator Sim is realized using the simulated signing algorithm and by reprogramming the ROM in the following way. This idea is taken from [Erw+21] but generalized to the notion of simulatability. We denote H to be the state of the random oracle. Note that we assume in Lemma 17 the existence of a function V_0 that on input σ outputs the one-way function of the randomness $\mathsf{OWF}(r)$. The algorithm Sim saves the state of the random oracle in

```
$\mathsf{Sim}^{\mathcal{O}_{\mathsf{Sign}},H}(m, Y)$
1 : $H' := H$
2 : $\sigma \leftarrow \mathcal{O}_{\mathsf{Sign}}(m)$
3 : $\mathsf{OWF}(r) \leftarrow V_0(\mathsf{simpk}, \sigma)$
4 : if $H'[\mathsf{simpk}||\mathsf{OWF}(r)||m] \neq \perp \vee H'[\mathsf{simpk}||\mathsf{OWF}(r) \cdot Y||m] \neq \perp$
5 :     abort
6 : $x := \mathsf{simpk}||\mathsf{OWF}(r) \cdot Y||m$
7 : $H[\mathsf{simpk}||\mathsf{OWF}(r)||m] := H[x]$
8 : $H[x] \leftarrow\$ \mathsf{ChSet}$
9 : return $\sigma$
```

Figure 22: The Sim algorithm of the transparent reduction of ID-based signature schemes.

the variable H'. Then, it obtains a valid signature from the signing oracle on the message m. Then it checks if the old random oracle was already queried either on the input $\mathsf{simpk}||\mathsf{OWF}(r)||m$ or on the input $\mathsf{simpk}||\mathsf{OWF}(r) \cdot Y||m$. If this is the case, it aborts. Note that by the size of the randomness space $\mathcal{D}_{\mathsf{R}}$, this happens only with negligible probability. Then it reprograms the random oracle, such that the signature σ becomes a pre-signature by swapping the inputs of $H[\mathsf{simpk}||\mathsf{OWF}(r)||m]$ and $H[\mathsf{simpk}||\mathsf{OWF}(r) \cdot Y||m]$. Finally, Sim returns the value σ. As the ROM was never queried on these values, no distinguisher can tell the distribution of "normal" pre-signing and the output of Sim apart. □

Acknowledgments

We gratefully acknowledge the anonymous reviewers for their invaluable comments and suggestions. We are also especially thankful to Paul Gahman for his dedicated efforts in enhancing the editorial quality of this work. This work was partially supported by Deutsche Forschungsgemeinschaft as part of the Research and Training Group 2475 "Cybercrime and Forensic Computing" (grant number 393541319/GRK2475/1-2019), grant 442893093, by the state of Bavaria at the Nuremberg Campus of Technology (NCT) which is a research cooperation between the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Technische Hochschule Nürnberg Georg Simon Ohm (THN), and by the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation program in the scope of the CONFIDENTIAL6G project under Grant Agreement 101096435, The contents of this publication are the sole responsibility of the authors and do not in any way reflect the views of the EU.

References

[Alb+22] Martin R. Albrecht et al. "Lattice-Based SNARKs: Publicly Verifiable, Preprocessing, and Recursively Composable". In: *Advances in Cryptology – CRYPTO 2022*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Cham: Springer Nature Switzerland, 2022, pp. 102–132. ISBN: 978-3-031-15979-4.

[ASM06] Man Ho Au, Willy Susilo, and Yi Mu. "Constant-Size Dynamic k-TAA". In: *SCN 06: 5th International Conference on Security in Communication Networks*. Ed. by Roberto De Prisco and Moti Yung. Vol. 4116. Lecture Notes in Computer Science. Maiori, Italy: Springer, Heidelberg, Germany, 2006, pp. 111–125. DOI: 10.1007/11832072_8.

[Aum+21] Lukas Aumayr et al. "Generalized Channels from Limited Blockchain Scripts and Adaptor Signatures". In: *Advances in Cryptology – ASIACRYPT 2021, Part II*. Ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13091. Lecture Notes in Computer Science. Singapore: Springer, Heidelberg, Germany, 2021, pp. 635–664. DOI: 10.1007/978-3-030-92075-3_22.

[BB04] Dan Boneh and Xavier Boyen. "Short Signatures Without Random Oracles". In: *Advances in Cryptology – EUROCRYPT 2004*. Ed. by Christian Cachin and Jan Camenisch. Vol. 3027. Lecture Notes in Computer Science. Interlaken, Switzerland: Springer, Heidelberg, Germany, 2004, pp. 56–73. DOI: 10.1007/978-3-540-24676-3_4.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. "Short Group Signatures". In: *Advances in Cryptology – CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, 2004, pp. 41–55. DOI: 10.1007/978-3-540-28628-8_3.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. "Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)". In: *20th Annual ACM Symposium on Theory of Computing*. Chicago, IL, USA: ACM Press, 1988, pp. 103–112. DOI: 10.1145/62212.62222.

[BL09] Ernie Brickell and Jiangtao Li. *Enhanced Privacy ID from Bilinear Pairing*. Cryptology ePrint Archive, Report 2009/095. https://eprint.iacr.org/2009/095. 2009.

[Bon+03] Dan Boneh et al. "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps". In: *Advances in Cryptology – EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. Lecture Notes in Computer Science. Warsaw, Poland: Springer, Heidelberg, Germany, 2003, pp. 416–432. DOI: 10.1007/3-540-39200-9_26.

[BR93] Mihir Bellare and Phillip Rogaway. "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols". In: *ACM CCS 93: 1st Conference on Computer and Communications Security*. Ed. by Dorothy E. Denning et al. Fairfax, Virginia, USA: ACM Press, 1993, pp. 62–73. DOI: 10.1145/168588.168596.

[BSW06] Dan Boneh, Emily Shen, and Brent Waters. "Strongly Unforgeable Signatures Based on Computational Diffie-Hellman". In: *PKC 2006: 9th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Moti Yung et al. Vol. 3958. Lecture Notes in Computer Science. New York, NY, USA: Springer, Heidelberg, Germany, 2006, pp. 229–240. DOI: 10.1007/11745853_15.

[CDL16a] Jan Camenisch, Manu Drijvers, and Anja Lehmann. *Anonymous Attestation Using the Strong Diffie Hellman Assumption Revisited*. Cryptology ePrint Archive, Report 2016/663. https://eprint.iacr.org/2016/663. 2016.

[CDL16b] Jan Camenisch, Manu Drijvers, and Anja Lehmann. "Anonymous Attestation Using the Strong Diffie Hellman Assumption Revisited". In: *Trust and Trustworthy Computing*. Ed. by Michael Franz and Panos Papadimitratos. Cham: Springer International Publishing, 2016, pp. 1–20. ISBN: 978-3-319-45572-3.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. "The Random Oracle Methodology, Revisited". In: *J. ACM* 51.4 (2004), 557–594. ISSN: 0004-5411. DOI: 10.1145/1008731.1008734. URL: https://doi.org/10.1145/1008731.1008734.

[CL03] Jan Camenisch and Anna Lysyanskaya. "A Signature Scheme with Efficient Protocols". In: *SCN 02: 3rd International Conference on Security in Communication Networks*. Ed. by Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano. Vol. 2576. Lecture Notes in Computer Science. Amalfi, Italy: Springer, Heidelberg, Germany, 2003, pp. 268–289. DOI: 10.1007/3-540-36413-7_20.

[DH76] W. Diffie and M. Hellman. "New directions in cryptography". In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638.

[DOY22] Wei Dai, Tatsuaki Okamoto, and Go Yamamoto. "Stronger Security and Generic Constructions for Adaptor Signatures". In: *Progress in Cryptology – INDOCRYPT 2022*. Ed. by Takanori Isobe and Santanu Sarkar. Cham: Springer International Publishing, 2022, pp. 52–77. ISBN: 978-3-031-22912-1.

[DW15] Christian Decker and Roger Wattenhofer. "A Fast and Scalable Payment Network with Bitcoin Duplex Micropayment Channels". In: *Stabilization, Safety, and Security of Distributed Systems*. Ed. by Andrzej Pelc and Alexander A. Schwarzmann. Cham: Springer International Publishing, 2015, pp. 3–18. ISBN: 978-3-319-21741-3.

[Eck+20] Lisa Eckey et al. *Splitting Payments Locally While Routing Interdimensionally*. Cryptology ePrint Archive, Report 2020/555. https://eprint.iacr.org/2020/555. 2020.

[EEE20] Muhammed F. Esgin, Oguzhan Ersoy, and Zekeriya Erkin. "Post-Quantum Adaptor Signatures and Payment Channel Networks". In: *ESORICS 2020: 25th European Symposium on Research in Computer Security, Part II*. Ed. by Liqun Chen et al. Vol. 12309. Lecture Notes in Computer Science. Guildford, UK: Springer, Heidelberg, Germany, 2020, pp. 378–397. DOI: 10.1007/978-3-030-59013-0_19.

[Erw+21] Andreas Erwig et al. "Two-Party Adaptor Signatures from Identification Schemes". In: *PKC 2021: 24th International Conference on Theory and Practice of Public Key Cryptography, Part I*. Ed. by Juan Garay. Vol. 12710. Lecture Notes in Computer Science. Virtual Event: Springer, Heidelberg, Germany, 2021, pp. 451–480. DOI: 10.1007/978-3-030-75245-3_17.

[Fin] *Finema: Enterprise Decentralized Identity*. https://finema.co.

[Fis05] Marc Fischlin. "Communication-Efficient Non-interactive Proofs of Knowledge with Online Extractors". In: *Advances in Cryptology – CRYPTO 2005*. Ed. by Victor Shoup. Vol. 3621. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, 2005, pp. 152–168. DOI: 10.1007/11535218_10.

[FS10] Marc Fischlin and Dominique Schröder. "On the Impossibility of Three-Move Blind Signature Schemes". In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. Lecture Notes in Computer Science. French Riviera: Springer, Heidelberg, Germany, 2010, pp. 197–215. DOI: 10.1007/978-3-642-13190-5_10.

[Gla+22] Noemi Glaeser et al. "Foundations of Coin Mixing Services". In: *ACM CCS 2022: 29th Conference on Computer and Communications Security*. Ed. by Heng Yin et al. Los Angeles, CA, USA: ACM Press, 2022, pp. 1259–1273. DOI: 10.1145/3548606.3560637.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L Rivest. "A digital signature scheme secure against adaptive chosen-message attacks". In: *SIAM Journal on computing* 17.2 (1988), pp. 281–308.

[GQ90] Louis C. Guillou and Jean-Jacques Quisquater. “A “Paradoxical” Indentity-Based Signature Scheme Resulting from Zero-Knowledge”. In: *Advances in Cryptology – CRYPTO’88*. Ed. by Shafi Goldwasser. Vol. 403. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, 1990, pp. 216–231. DOI: `10.1007/0-387-34799-2_16`.

[HK09] Dennis Hofheinz and Eike Kiltz. “The Group of Signed Quadratic Residues and Applications”. In: *Advances in Cryptology – CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, 2009, pp. 637–653. DOI: `10.1007/978-3-642-03356-8_37`.

[Hyp] *Hyperledger Ursa*. `https://github.com/hyperledger/ursa`.

[JB09] Mahabir Prasad Jhanwar and Rana Barua. “Sampling from Signed Quadratic Residues: RSA Group Is Pseudofree”. In: *Progress in Cryptology - INDOCRYPT 2009: 10th International Conference in Cryptology in India*. Ed. by Bimal K. Roy and Nicolas Sendrier. Vol. 5922. Lecture Notes in Computer Science. New Delhi, India: Springer, Heidelberg, Germany, 2009, pp. 233–247.

[KMP16] Eike Kiltz, Daniel Masny, and Jiaxin Pan. “Optimal Security Proofs for Signatures from Identification Schemes”. In: *Proceedings, Part II, of the 36th Annual International Cryptology Conference on Advances in Cryptology — CRYPTO 2016 - Volume 9815*. Berlin, Heidelberg: Springer-Verlag, 2016, 33–61. ISBN: 9783662530078. DOI: `10.1007/978-3-662-53008-5_2`. URL: `https://doi.org/10.1007/978-3-662-53008-5_2`.

[KW03] Jonathan Katz and Nan Wang. “Efficiency Improvements for Signature Schemes with Tight Security Reductions”. In: *ACM CCS 2003: 10th Conference on Computer and Communications Security*. Ed. by Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger. Washington, DC, USA: ACM Press, 2003, pp. 155–164. DOI: `10.1145/948109.948132`.

[Loo+23] Tobias Looker et al. *The BBS Signature Scheme*. Internet-Draft draft-irtf-cfrg-bbs-signatures-02. Work in Progress. Internet Engineering Task Force, Mar. 2023. 71 pp. URL: `https://datatracker.ietf.org/doc/draft-irtf-cfrg-bbs-signatures/02/`.

[Mad+22] Varun Madathil et al. *Cryptographic Oracle-Based Conditional Payments*. Cryptology ePrint Archive, Paper 2022/499. `https://eprint.iacr.org/2022/499`. 2022. DOI: `10.14722/ndss.2023.23024`. URL: `https://eprint.iacr.org/2022/499`.

[Mad+23] Varun Madathil et al. “Cryptographic Oracle-based Conditional Payments”. In: *Proceedings 2023 Network and Distributed System Security Symposium* (2023).

[Mal+19] Giulio Malavolta et al. “Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability”. In: *ISOC Network and Distributed System Security Symposium – NDSS 2019*. San Diego, CA, USA: The Internet Society, 2019.

[Mil+19] Andrew Miller et al. “Sprites and State Channels: Payment Networks that Go Faster Than Lightning”. In: *FC 2019: 23rd International Conference on Financial Cryptography and Data Security*. Ed. by Ian Goldberg and Tyler Moore. Vol. 11598. Lecture Notes in Computer Science. Frigate Bay, St. Kitts and Nevis: Springer, Heidelberg, Germany, 2019, pp. 508–526. DOI: `10.1007/978-3-030-32101-7_30`.

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

[Qin+23] Xianrui Qin et al. “BlindHub: Bitcoin-Compatible Privacy-Preserving Payment Channel Hubs Supporting Variable Amounts”. In: *2023 IEEE Symposium on Security and Privacy (SP)*. 2023, pp. 2462–2480. DOI: `10.1109/SP46215.2023.10179427`.

[RS09] Markus Rückert and Dominique Schröder. “Security of Verifiably Encrypted Signatures and a Construction without Random Oracles”. In: *PAIRING 2009: 3rd International Conference on Pairing-based Cryptography*. Ed. by Hovav Shacham and Brent Waters. Vol. 5671. Lecture Notes in Computer Science. Palo Alto, CA, USA: Springer, Heidelberg, Germany, 2009, pp. 17–34. DOI: `10.1007/978-3-642-03298-1_2`.

[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`.

[Sho97] Victor Shoup. “Lower Bounds for Discrete Logarithms and Related Problems”. In: *Advances in Cryptology – EUROCRYPT’97*. Ed. by Walter Fumy. Vol. 1233. Lecture Notes in Computer Science. Konstanz, Germany: Springer, Heidelberg, Germany, 1997, pp. 256–266. DOI: `10.1007/3-540-69053-0_18`.

[TMM20] Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. *Post-Quantum Adaptor Signature for Privacy-Preserving Off-Chain Payments*. Cryptology ePrint Archive, Report 2020/1345. `https://eprint.iacr.org/2020/1345`. 2020.

[TMSS23] Erkan Tairi, Pedro Moreno-Sanchez, and Clara Schneidewind. *LedgerLocks: A Security Framework for Blockchain Protocols Based on Adaptor Signatures*. Cryptology ePrint Archive, Paper 2023/1315. `https://eprint.iacr.org/2023/1315`. 2023. URL: `https://eprint.iacr.org/2023/1315`.

[Tsa+07] Patrick P. Tsang et al. “Blacklistable anonymous credentials: blocking misbehaving users without ttps”. In: *ACM CCS 2007: 14th Conference on Computer and Communications Security*. Ed. by Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson. Alexandria, Virginia, USA: ACM Press, 2007, pp. 72–81. DOI: `10.1145/1315245.1315256`.

[Wat05] Brent R. Waters. “Efficient Identity-Based Encryption Without Random Oracles”. In: *Advances in Cryptology – EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. Lecture Notes in Computer Science. Aarhus, Denmark: Springer, Heidelberg, Germany, 2005, pp. 114–127. DOI: `10.1007/11426639_7`.

Supplementary Materials

A Definitions

A.1 Digital Signatures

In a digital signature scheme [DH76; GMR88], a signer runs the signing algorithm Sign to generate a signature σ on a message m using his private key sk. The signature's validity can be publicly verified with the algorithm Vrfy, which requires the signer's public verification key vk, a message m, and a signature σ.

Definition of Signatures. We recall the definition of digital signatures.

Definition 33 (Digital signature). *Let $\mathcal{D}_{\mathsf{K}}$ be the secret key space, $\mathcal{D}_{\mathsf{K'}}$ be the public key space, $\mathcal{D}_{\mathsf{R}}$ be the randomness space, $\mathcal{D}_{\Sigma}$ be the signature space and $\mathcal{D}_{\mathsf{M}}$ be the message space. A* digital signature scheme $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *for messages of length ℓ_m is a triple of efficient algorithms defined as:*
$\underline{(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(\lambda).}$ *The* key generation *algorithm* $\mathsf{KGen}(1^\lambda)$ *is a* PPT *algorithm that on input a security parameter 1^λ, outputs a key pair $(\mathsf{sk}, \mathsf{vk}) \in \mathcal{D}_{\mathsf{K}} \times \mathcal{D}_{\mathsf{K'}}$.*
$\underline{\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m).}$ *The input of the* PPT signing *algorithm* $\mathsf{Sign}(\mathsf{sk}, m)$ *is a private key* $\mathsf{sk} \in \mathcal{D}_{\mathsf{K}}$ *and message $m \in \{0,1\}^{\ell_m} = \mathcal{D}_{\mathsf{M}}$, it outputs a signature $\sigma \in \mathcal{D}_{\Sigma}$.*
$\underline{b \leftarrow \mathsf{Vrfy}(\mathsf{vk}, m, \sigma).}$ *The* verification *algorithm* $\mathsf{Vrfy}(\mathsf{vk}, m, \sigma)$ *is a* DPT *algorithm that takes as input a a public key $\mathsf{vk} \in \mathcal{D}_{\mathsf{K'}}$, a message $m \in \{0,1\}^{\ell_m} = \mathcal{D}_{\mathsf{M}}$ and a signature $\sigma \in \mathcal{D}_{\Sigma}$ and outputs a bit b.*

We define the correctness of signatures in the usual way:

Definition 34 (Correctness). *A digital signature scheme Σ is* correct *if for any $m \in \{0,1\}^{\ell_m}$ it holds, that for all $\lambda \in \mathbb{N}$ and all $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$:*

$$\Pr[\mathsf{Vrfy}(\mathsf{vk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1,$$

where the randomness is taken over the random choice of KGen *and* Sign.

Security of Signatures. In the following, we recall the notions of unforgeability and strong unforgeability. In both games, the adversary $\mathcal{A}$ is given as input the public vk and has access to a signing oracle $\mathcal{O}_{\mathsf{Sign}}(\cdot)$ that $\mathcal{A}$ may query adaptively on messages of his choice. The difference between unforgeability and strong unforgeability lies in the success determination of $\mathcal{A}$. In case of unforgeability, the adversary wins the game if it computes a valid signature σ^* on a new message m^*, i.e., one that was not submitted to the signing oracle. The notion of strong unforgeability demands that $\mathcal{A}$ must compute a fresh message-signature pair (m^*, σ^*) to win the game. The difference is that $\mathcal{A}$ can win the game even if it computes a new signature for a message it queried to the signing oracle. Both security notions are defined in the following. We mark the slight differences between both security definitions with a gray background.

Definition 35 (EUF-CMA (SUF-CMA) security)**.** *A signature scheme Σ is (strongly) existential unforgeable under a chosen message attack or* EUF-CMA *resp.* (SUF-CMA) *secure, if for every* PPT *adversary $\mathcal{A}$ there exists a negligible function ν such that for all $\lambda in \mathbb{N}$,* $\Pr\big[\mathsf{SigForge}_{\mathcal{A},\Sigma}(\lambda) = 1\big] \le \nu(\lambda)$*, where the experiment* $\mathsf{SigForge}_{\mathcal{A},\Sigma}$ *is defined in Figure 23 and the probability is taken over all random choices of all randomized algorithms.*

```
$\mathsf{SigForge}_{\mathcal{A},\Sigma}(\lambda)$
1 : $\mathcal{Q} := \emptyset$
2 : $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(\mathsf{vk})$
4 : return $(m^* \notin \mathcal{Q} \wedge \mathsf{Vrfy}(\mathsf{vk}, m^*, \sigma^*))$
5 : return $((m^*, \sigma^*) \notin \mathcal{Q} \wedge \mathsf{Vrfy}(\mathsf{vk}, m^*, \sigma^*))$
```

```
$\mathcal{O}_{\mathsf{Sign}}(\mathsf{sk}, m)$
1 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
2 : $\mathcal{Q} = \mathcal{Q} \cup \{m\}$
3 : $\mathcal{Q} = \mathcal{Q} \cup \{(m, \sigma)\}$
4 : return $\sigma$
```

Figure 23: Security notions of digital signatures, where the gray part refers to strong unforgeability and the regular part to unforgeability. The gray winning condition replaces the line before.

B The Camenisch-Lysyanskaya Signature Scheme

The Camenisch-Lysyanskaya signature scheme is a signature scheme [CL03] based on the strong RSA assumption on a flexible RSA instance and proven secure without using the random oracle model. Therefore, it is not an identification-based signature scheme. The CL signature scheme as given in [CL03] only achieves EUF-CMA security. So to the original CL signature scheme, Theorem 4 can not be applied. This is given through the fact that one can rerandomize a given valid signature to get another in the following way: Given a valid signature, message pair $(\sigma, m) = (v, e, r, m)$. One can choose a random element $r' \leftarrow \mathbb{Z}_{\ell_r}$ and compute $r^* = r' \cdot e$. Then the signature, message pair $(\sigma^*, m) = (v \cdot b^{r'}, e, r + r^*, m)$ is valid as $\big(v \cdot b^{r'}\big)^e = v^e b^{r'e} \equiv a^m b^r b^{r^*} c \equiv a^m b^{r+r^*} c \bmod n$. To avoid this, we can restrict the prime number e to be bigger than the random number r. Choosing $\ell_e \ge l_r + 2$, in fact, leads to proof for the SUF-CMA security of the CL signature scheme, as we will show in Lemma 15.

Furthermore, we extend the key of the CL^+ signature scheme to obtain a signature scheme with a simulatable transparent reduction. For this purpose, we use the one-way function $\mathsf{OWF} : \mathcal{D}_{\mathsf{R}} \to \mathcal{D}_{\mathsf{R}'}$ that maps x to b^x to build a hard relation. The simulation of pre-sign queries needs to compute the e-th root of the statement $Y = b^y$ for a random prime e, even if this is hard in our RSA group without knowing the factors of n. We circumvent this problem by providing the auxiliary information $\mathsf{aux} = u^y$ to the statement Y, such that there exists an element s, for which $u^s = b$. So now if e divides s and we know an element x, such that $x \cdot e = s$, we can compute $\mathsf{aux}^x = u^{yx} = u^{s/e \cdot y} = b^{1/e \cdot y} = Y^{1/e}$ without knowing the witness y. To check if the auxiliary information is well computed, we can check if $Y = \mathsf{aux}^s$. To allow any party to compute this auxiliary information, we append the value u from the secret key to the public parameters of the hard relation.

Furthermore, the secret key holder of the CL^+ signature scheme can efficiently check if auxiliary information is well-formed since it knows s, such that $b = u^s$.
We now prove the following: The modified Camenisch-Lysyanskaya signature scheme CL^+ (Definition 29) achieves SUF-CMA security (Definition 35) under the Strong RSA assumption (Definition 39).

Proof of Lemma 15 . This proof is taken from [CL03] almost verbatim. It remained to add some argumentation to the proof of Lemma 5 from the original paper to handle forgeries of Type 2 that are signatures on the same message. Additionally, we added Type 4 forgeries. Furthermore, we combined the three proofs into one reduction.

To prove this theorem, we want to reduce the SUF-CMA security of the signature scheme to the hardness of the strong RSA assumption on a flexible RSA instance. Therefore, we assume that an adversary $\mathcal{A}$ exists that breaks the SUF-CMA security of the scheme with non-negligible probability $\epsilon(\lambda)$. We then create a reduction $\mathcal{R}$ that, on input an instance of the flexible RSA problem (n, u), simulates the StrongSigForge game to this adversary and uses the adversaries ability to break the SUF-CMA security to break the strong RSA assumption on the flexible RSA instance (n, u). The reduction computes a public key vk that is indistinguishable from a soundly computed public key and provides a signing oracle to the adversary $\mathcal{A}$. When the adversary outputs its forgery, the reduction $\mathcal{R}$ uses this forgery to break the strong RSA assumption.

We now assume that we know the number K of the adversary's oracle queries. We can do so because otherwise, we could estimate this number experimentally. Furthermore, with the Markov inequality, half the time, $\mathcal{A}$ uses less than $2K$ oracle queries. The forgery σ^* given from the adversary has the following form (e, v, r). Since the forgery is a new pair (m^*, σ^*), there are four possible types of forgeries:

Type 1 The prime number e from the forgery was never returned by the signing oracle.

Type 2 The prime number e was previously returned from the signing oracle at query i, but the root v is different from the returned root v_i.

Type 3 The prime number e was previously returned from the signing oracle at query i, and the root v equals the root v_i, but the tuple (m, s) is new.

Type 4 The prime number e was never returned by the signing oracle, but e divides s.

On input of the flexible RSA instance (n, u), the reduction $\mathcal{R}$ guesses if the provided forgery will be of even or odd type and chooses the matching algorithms. It provides the signing oracle to the adversary using these algorithms.

Key Generation(odd) Choose $2K$ random primes $e_1, \dots e_{2K}$ of size ℓ_e. Choose, at random values $r_1, r_2 \in \mathbb{Z}_{n^2}$. Let $a = u^E$, where $E = \prod_{i=1}^{2K} e_i$, $b = a^{r_1}$, $c = a^{r_2}$, $s = E \cdot r_1$. Let (n, a, b, c) be the public key.

Key Generation(even) Choose $2K$ random primes $e_1, \dots e_{2K}$ of size ℓ_e. Choose, at random values $\alpha, \beta \in \mathbb{Z}_{n^2}$. Choose a random value t of length ℓ_r. Let $b = u^E$, where $E = \prod_{j=1}^{2K} e_j / e_i$, $a = b^\alpha \bmod n$, $c = b^{e_i \beta - t}$, $s = E$. Let (n, a, b, c) be the public key.

Signing(odd) Receiving the i^{th} signing query m_i, choose a value r_i of length ℓ_r uniformly at random. Compute $v_i = a_i^{m_i} b_i^{r_i} c_i$ with $a_i = u^{E/e_i}$, $b_i = a_i^{r_1}$, $c_i = a_i^{r_2}$ and return (e_i, v_i, r_i).

Signing(even) Receiving the j^{th} signing query m_j with $j \neq i$, choose a value r_j of length ℓ_r uniformly at random. Compute $v_j = a_j^{m_j} b_j^{r_j} c_j$ with $b_j = u^{E/e_j}$, $a_j = b_j^{\alpha}$, $c_j = b_j^{e_i\beta - t}$ and return (e_j, v_j, r_j). Receiving the i^{th} signature query m_i, compute the value $r_i = t - \alpha m_i$ and $v_i = b^{\beta}$. Return (e_i, v_i, r_i). It holds, that $v_i^{e_i} = b^{e_i\beta + t - t} = a^{m_i} b^{r_i} c$.

Note that the reduction's public key and all the signatures provided have the same distribution as in the original scheme. When the adversary provides its message, forgery pair (m^*, σ^*) that is valid, the reduction aborts if the guess of even or odd was incorrect. This happens with probability at most 1/4 by a standard hybrid argument. The following break algorithms show how the strong RSA assumption on the flexible RSA problem can be broken given a forgery of the particular type when the guess of odd or even was correct. This is done if the break algorithms output values (v, e) such that $e > 1$ and $v^e \equiv u \mod n$. The algorithms get as input from the key generation algorithm all the primes $e_1, \ldots, e_{2K}$ and the values α, β, r_1, r_2 and from the reduction of the forgery, message pair (m, e, v, r) as well as the queue $\mathcal{Q} = \{(m_i, \sigma_i)\}$ and the instance of the flexible RSA instance (n, u).
Break(Type 1): The forgery (e, v, r) on message m with $e > 4$ gives us $v^e = a^m b^r c = u^{E(m + r_1 r + r_2)}$. Note that for a Type 1 forgery, $\gcd(E, e) = 1$. If $E(m + r_1 r + r_2)$ and e are relatively prime, we can use Lemma 19 (Shamir's Trick) to break the strong RSA assumption. We will use the following claim to use Shamir's Trick: With probability at least 1/2 over the random choices made by the reduction, it is the case that either $\gcd(e, m + r_1 r + r_2) < e$, or, on input e, one can efficiently factor n. This Claim is proven in [CL03]. Therefore, Break(Type 1) breaks the strong RSA assumption whenever the number of queries does not exceed $2K$, which happens with probability 1/2 by the Markov inequality, and whenever the conditions of the claim are satisfied, which occurs with probability at least 1/2. So Break(Type 1) succeeds with probability at least $\frac{1}{4}$.
Break(Type 3): For the forgery signature (m, e, v, r) and the i^{th} signature (m_i, e_i, v_i, r_i) it holds, that $v = v_i$ and $e = e_i$, but the pair (m, r) is new. With those equalities, it holds, that $a^m b^r \equiv a^{m_i} b^{r_i}$. This implies that $m + r_1 r \equiv m_i + r_1 r_i \mod \phi(n)$. As $(m, r) \neq (m_i, r_i)$ and $r_1 > m, r_1 > m_i$, $m + r_1 r \neq m_i + r_1 r_i$. Therefore, $\phi(n) | m + r_1 r - m_i - r_1 r_i \neq 0$, and so by Corollary 1, this forgery allows the reduction to break the strong RSA assumption. The probability that Break(Type 3) succeeds is 1/2 due to the Markov inequality.
Break(Type 2): For the forgery signature (m, e, v, r) and the i^{th} signature (m_i, e_i, v_i, r_i) it holds, that $v \neq v_i$ and $e = e_i$. As both signatures verify, we have

$$v^{e_i} = a^m b^r c = b^{m\alpha + r + e_i\beta - t} = b^{(m - m_i)\alpha + (s - s_i) + e_i\beta}.$$

If $m \neq m_i$, this forgery is shown to break strong RSA in [CL03] with probabilty $1/4 \cdot \frac{1}{2K} = \frac{1}{8K}$.
If $m = m_i$, it holds that $r \neq r_i$, because else $v^e \equiv v_i^e$ with $v^e \neq v_i^e$. It is true that $v^e \neq v_i^e$ as $v_i \neq v$(Type 2), and the function that maps x to x^e in the group QR_n^+ is injective if e is relatively prime to the elements p', q' that generate n. And e is relatively prime because

otherwise, one could factorize n with the element e. Besides of this inequality, we have $e_i|(m-m_i)\alpha+(r-r_i)+e_i\beta = (r-r_i)+e_i\beta$ since both signatures, the provided one and the forgery, verify. This equation holds if and only if $e_i|r-r_i$. This can not be since $r \neq r_i$ and $l_e > l_r + 2$. Therefore, we have that $e_i \nmid (m\alpha + s + e_i\beta - t) = \gamma$ and thus we have $v_i^e \equiv u^{E\gamma}$, where $\gcd(e,\gamma) = 1$. This is the same setup as if m were not equal to m_i. So this breaks the strong RSA assumption as shown in [CL03]. Thus with probability 1/2 (Markov) $\cdot\frac{1}{2K}$ (right guess) $= \frac{1}{4K}$ Break(Type 2) succeeds if $m \neq m_i$.
Break(Type 4): If the forgery is of type four, then the reduction aborts. Note that this is only the case if the adversary found a prime that was never returned by the signing oracle but was selected to compute s. The probability of finding such an e is negligible due to the size of ℓ_e and the fact that factoring is hard in the RSA group.

The reduction does not abort with a probability of $> 1/4$ if it found the right forgery number, and if the forgery is of type four, it only aborts with negligible probability. If it does not abort, it can determine the type of the forgery; the forgery was not of type four and wins with non-negligible probability in each kind if the forgery verifies. Thus, the reduction breaks strong RSA with overwhelming probability if the adversary succeeds. And therefore, such an adversary cannot exist. □

Definition 36 (Camenisch Lysyanskaya Signature Scheme)**.** *Let $\lambda \in \mathbb{N}$ be the security parameter, $\ell_m, \ell_n, \ell_r \in \mathbb{N}$. The Camenisch Lysyanskaya signature scheme for the message space $\mathcal{D}_\mathsf{M} = \{0,1\}^{\ell_m}$ consists of the following algorithms:*

Key Generation $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$**.** *The key generation algorithm chooses a special RSA modulus $n = pq, p = 2p'+1, q = 2q'+1$ of the length $\ell_n = 2\lambda$. Furthermore, it chooses $a, b, c \in QR_n$ uniformly at random and outputs $\mathsf{vk} = (n,a,b,c)$ and $\mathsf{sk} = p$.*

Signing $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$**.** *The signing algorithm chooses a random prime number e of length $\ell_e \geq \ell_m + 2$, and a random number r of length $\ell_r = \ell_n + \ell_m + l$ where l is a security parameter. Finally, it computes a v such that $v^e \equiv a^m b^r c \bmod n$. It returns the triple (e, v, r).*

Verification $b \leftarrow \mathsf{Vrfy}(\mathsf{vk}, m, \sigma)$**.** *The verification algorithm verifies that the triple (e,v,r) is a signature $m \in \mathcal{D}_\mathsf{M}$ by checking that $v^e \equiv a^m b^r c \bmod n$ and checking that $2^{\ell_e} > e > 2^{\ell_e - 1}$.*

Theorem 6. *The Camenisch Lysyanskaya signature scheme for Definition 36 is dichotomic.*

Proof of Theorem 6. To show that CL signatures are dichotomic, we want to stress that for a given b that is part of the public key vk, the function $\mathsf{OWF} : \mathcal{D}_\mathsf{R} \to \mathcal{D}_{\mathsf{R}'}$ that maps x to b^x is a homomorphic function.
We firstly provide the following definitions for Σ_1 and Σ_2:
We define Σ_1 as $\Sigma_1(\mathsf{sk}, m; r) := r$. It is clear, that $\Sigma_1(\mathsf{sk}, m; r) + y = r + y = \Sigma_1(\mathsf{sk}, m; r + y)$. Furthermore we define Σ_2 as $\Sigma_2(\mathsf{sk}, m; r) = (v, e)$ that are defined as in Definition 29. Thus e is a random prime number and it holds, that $v^e \equiv a^m b^r c \bmod n$.
To show the verification property:
The verification of a signature (v,e,r) w.r.t. a message m is done via checking the equation $v^e \equiv a^m b^r c \bmod n$. This equation can be checked upon the inputs $((n,a,b,c),(v,e),b^r) = (\mathsf{vk}, \sigma_1, \mathsf{OWF}(\sigma_2))$. □

B.1 Number-Theoretic Basics of CL Signatures

The following definitions, lemmas, and corollaries are taken almost verbatim from the work of Camenisch and Lysyanskaya [CL03].

Definition 37 (Safe primes). *A prime number p is called safe if $p = 2p' + 1$, such that p' is also a prime number.*

Definition 38 (Special RSA modulus). *An RSA modulus $n = pq$ is special if $p = 2p' + 1$ and $q = 2q' + 1$ are safe primes.*

Definition 39 (Strong RSA Assumption). *The strong RSA assumption is that it is hard, on input a RSA modulus n and an element $u \in \mathbb{Z}_n^*$, to compute values $e > 1$ and v such that $v^e \equiv u \bmod n$. More formally, we assume that for all polynomial-time circuit families $\{\mathcal{A}_k\}$, there exists a negligible function $\nu(\lambda)$ such that*

$$Pr[n \leftarrow \mathsf{RSAmodulus}(1^\lambda); u \leftarrow \mathbb{Z}_n^*(v, e) \leftarrow \mathcal{A}_k(n, u) : e > 1 \wedge v^e \equiv u \ mod\ n] \leq \nu(\lambda).$$

The tuple (n, u) generated as above is called a *general instance* of the *flexible* RSA problem. By $QR_n \subseteq \mathbb{Z}_n^*$ we will denote the set of quadratic residues modulo n, i.e., elements $a \in \mathbb{Z}_n^*$ such that $\exists b \in \mathbb{Z}_n^*$ such that $b^2 \equiv a \bmod n$.

Lemma 18. *If $n = pq$, $p = 2p' + 1$, $q = 2q' + 1$ is a special RSA modulus, then QR_n is a cyclic group under multiplication of size $p'q'$, where all but $p' + q'$ of the elements are generators.*

Due to the above lemma, we can assume that a randomly chosen element of QR_n is a generator if (n, u) is an instance of the flexible RSA problem, and n is a special RSA modulus.

Lemma 19 (Shamirs Trick). *Let an integer n be given. Suppose that we are given the values $u, v, \in \mathbb{Z}_n^*$ and $x, y \in \mathbb{Z}$, $gcd(x, y) = 1$ such that $v^x \equiv u^y \ mod\ n$. Then there is an efficient procedure to compute the value z such that $z^x \equiv u \ mod\ n$.*

Corollary 1. *Let n be an integer and $u \leftarrow\!\!\$\ \mathbb{Z}_n^*$. Let e such that $gcd(e, \phi(n)) = 1$ be given. Given any value x such that $\phi(n)|x$, one can efficiently compute v such that $v^e \equiv u \ mod\ n$.*

B.1.1 Signed Quadratic Residues

In the following, we recite results from Hofheinz and Kiltz [HK09]. In particular, we show how to apply the results from Appendix B.1 to the setting of signed quadratic residues.

Definition 40. *Let $n \in \mathbb{N}$. We define the* signed quadratic residues (QR_n^+) *as*

$$QR_n^+ := \{|x| : x \in QR_n\},$$

where $|x|$ is the absolute value when representing elements of $\mathbb{Z}_n$ as the set $\{-(n-1)/2, \ldots, (n-1)/2\}$.

We start by reciting the lemma, which shows that the signed quadratic residues are a cyclic, efficiently recognizable group of order $\phi(n)$.

Lemma 20. *Let n be a Blum integer (i.e. $n \equiv 3 \mod 4$). Then:*

1. *(QR_n^+) is a group of order $\phi(n)/4$.*
2. *QR_n^+ is efficiently recognizable (given only n).*
3. *If QR_n is cyclic, so is QR_n^+.*

We first observe that the strong RSA assumption is defined w.r.t. elements $u \in \mathbb{Z}_n^*$. Since QR_n^+ is a subgroup of $\mathbb{Z}_n^*$, the strong RSA assumption is applicable. The same holds for Shamirs Trick and Corollary 1. Secondly, if n is a special RSA modulus, then QR_n is a cyclic group. By Lemma 20, we know that the same holds for QR_n^*. In addition, each generator g of QR_n is also a generator of QR_n^+ when transformed to $|g|$. Henceforth, Lemma 18 is also applicable to QR_n^+. It remains to show that the strong RSA assumption w.r.t. the signed quadratic residues are at least as hard as the strong RSA assumption w.r.t. the quadratic residues (c.f. Definition 39). We do this by reciting a theorem of [JB09].

Theorem 7 (Theorem 1 of [JB09]). *If the strong QR-RSA problem is asymptotically hard where the underlying modulus n is chosen randomly from the set of all safe prime products of bit size bounded by λ, then the strong signed QR-RSA problem is also asymptotically hard for the same n.*

Since this theorem holds, from now on, we drop the corresponding subgroup $\mathbb{Z}_n^*$, QR_n, or QR_n^+ when referring to the strong RSA assumption.

C Additional Lemmas

Lemma 21. *Construction 1 satisfies pre-signature correctness and pre-signature adaptability.*

Proof of Pre-Signature Correctness. Let $m \in \{0,1\}^*$ be an arbitrary message and $\lambda \in \mathbb{N}$. Let $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(Y, y) \leftarrow \mathsf{RGen}(1^\lambda)$, $\widetilde{\sigma} \leftarrow \mathsf{pSign}_{\mathsf{sk}}(m, Y)$, $\sigma := \mathsf{Adapt}(\mathsf{vk}, (\widetilde{\sigma}, y))$ and $y' := \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)$. Let $\widetilde{\sigma} := (\sigma_1, \sigma_2)$. We first show that $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}) = 1$: By the construction, it holds, that $\mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma}) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2) \cdot Y)$. With the homomorphism of the function OWF and by the definition of the relation, we have that

$$\mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2) \cdot Y) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2 + y)),$$

since $Y = \mathsf{OWF}(y)$. With the definition of a dichotomic signature, $\sigma_1 = \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot Y)$ and $\sigma_2 + y = \Sigma_2(\mathsf{sk}, m, \sigma_1; r + y)$. Therefore,

$$\mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2 + y)) = \mathsf{Vrfy}'(\mathsf{vk}, m, \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r) \cdot Y), \mathsf{OWF}(\Sigma_2(\mathsf{sk}, m, \sigma_1; r) + y).$$

With the homomorphic property of OWF, this equals to

$$\mathsf{Vrfy}'(\mathsf{vk}, m, \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r + y)), \mathsf{OWF}(\Sigma_2(\mathsf{sk}, m, \sigma_1; \mathsf{OWF}(r + y))).$$

By the definition of dichotomic signatures, this equals to

$$\mathsf{Vrfy}(\mathsf{vk}, m, (\Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r + y)), \Sigma_2(\mathsf{sk}, m, \sigma_1, \mathsf{OWF}(r + y))$$

and with the correctness of the underlying signature scheme this equals to 1 since this is a valid signature for the message m and the randomness $r + y$.
Now we show that $\mathsf{Vrfy}(\mathsf{vk}, m, \sigma) = 1$: By the definition of Adapt it holds, that

$$\mathsf{Vrfy}(\mathsf{vk}, m, \sigma) = \mathsf{Vrfy}(\mathsf{vk}, m, (\sigma_1, \sigma_2 + y)).$$

The same computation as for pVrfy shows, that

$$\mathsf{Vrfy}(\mathsf{vk}, m, (\sigma_1, \sigma_2 + y)) = \mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2 + y)) =$$
$$\mathsf{Vrfy}(\mathsf{vk}, m, \Sigma_1(\mathsf{sk}, m; \mathsf{OWF}(r + y)), \Sigma_2(\mathsf{sk}, m, \Sigma_1; r + y) = 1.$$

Finally, we need to show that $(Y, y') \in \mathsf{Rel}$: By construction, $y' = (\sigma_2 + y) - \sigma_2 = y$. And $(Y, y) \in \mathsf{Rel}$ by definition. $\square$

Proof of Pre-Signature Adaptability. Let $m \in \{0,1\}^*$ be a arbitrary message, $\lambda \in \mathbb{N}$, $(Y, y) \in \mathsf{Rel}$ a hard relation, vk be a public key, $\widetilde{\sigma} \leftarrow \{0,1\}^*$ be a pre signature such that

$$\mathsf{pVrfy}(\mathsf{vk}, (m, Y, \widetilde{\sigma})) = 1.$$

This means that for a pre-signature $\widetilde{\sigma} = (\sigma_1, \sigma_2)$, it holds that $\mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2) \cdot Y) = 1$. By the homomorphic property of OWF, this equals to $\mathsf{Vrfy}'(\mathsf{vk}, m, \sigma_1, \mathsf{OWF}(\sigma_2 + y)) = 1$. And with the dichotomic definition this equals to $\mathsf{Vrfy}(\mathsf{vk}, m, \sigma_1, \sigma_2 + y)$ and this equals to $\mathsf{Vrfy}(\mathsf{vk}, m, \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y))$ since $\mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y) = \mathsf{Adapt}(\mathsf{vk}, (\sigma_1, \sigma_2), y) = (\sigma_1, \sigma_2 + y)$. So any pre-signature that verifies can be adapted to be a full signature that verifies. $\square$

To finish the proof of Theorem 4, it remains to show full extractability. The proof of full extractability depends on an auxiliary lemma, which we now state.

Lemma 22. *Let* $\mathsf{OWF} : \mathcal{D}_\mathsf{R} \to \mathcal{D}_{\mathsf{R}'}$ *be an injective homomorphic one-way function, and* Rel *be a canonical hard relation with auxiliary input for* OWF*. Let* Σ *be a dichotomic signature scheme with respect to* OWF *that has a simulatable transparent reduction* $\mathcal{T}$ *from the* $\mathsf{SUF\text{-}CMA}$ *security of* Σ *to an underlying hard problem* Π*. Then for any* $Y \in \mathcal{D}_{\mathsf{R}'}$ *and any message* $m \in \mathcal{D}_\mathsf{M}$ *and any* PPT *distinguisher* $\mathcal{D}$ *for the values* $\mathcal{S}_\mathsf{Sim} := \{\widetilde{\sigma} : \widetilde{\sigma} \leftarrow \mathcal{T}.\mathsf{Sim}(\mathsf{simsk}, m, Y)\}$ *and* $\mathcal{S}_\mathsf{pSign} := \{\widetilde{\sigma} : (\sigma_1, \sigma_2) \leftarrow \mathcal{T}.\mathsf{SimSign}(\mathsf{simsk}, m), \widetilde{\sigma} := (\sigma_1, \sigma_2 - y)\}$*, the equation*

$$|\Pr[\mathcal{D}(\mathcal{S}_\mathsf{Sim}, \lambda) = 1] - \Pr[\mathcal{D}(\mathcal{S}_\mathsf{pSign}, \lambda) = 1]| \leq \nu(\lambda)$$

holds, where the probabilities are derived from the random choices of m, Y, λ *and the random choices of the probabilistic algorithms.*

Proof. Note that an adaptor signature scheme of Construction 1 has the following property: A pre-signature $\widetilde{\sigma}$ for the message - randomness pair (m, r) and the statement - witness pair (Y, y) by definition is equal to $(\sigma_1, \sigma_2 - y)$ where (σ_1, σ_2) is the dichotomic signature for the message, randomness pair $(m, r + y)$. By the injectivity of OWF, both sets are equally distributed; therefore, no distinguisher between these two sets can exist. $\square$

D Proofs of the Counter Examples

This section shows that our counter-examples in Section 4 are secure adaptor signatures w.r.t. Definition 11.

Lemma 23. *Let* AS *be an adaptor signature scheme that is secure w.r.t. Definition 11 and achieves extractability. Then, the modified adaptor signature scheme* AS' *as for Fig. 5 is secure w.r.t. Definition 11.*

Proof. To prove Lemma 23, we show pre-signature correctness, pre-signature adaptability, and extractability. Since extractability implies unforgeability and witness extractability, AS' is secure w.r.t. Definition 11.
Pre-Signature Correctness. The pre-signature correctness of AS' follows directly from the pre-signature correctness of AS, since the algorithms pSign' and pVrfy' run the algorithms pSign and pVrfy twice. Moreover, Adapt' and $\mathsf{Extract}'$ run Adapt and $\mathsf{Extract}$ on the first pre-signature part, which is a AS pre-signature.
Pre-Signature Adaptability. Since the algorithm pVrfy' checks, if the first part of a pre-signature, namely $\widetilde{\sigma}_1$ pre-verifies under pVrfy, and the algorithms Adapt', and $\mathsf{Extract}'$ consider only $\widetilde{\sigma}_1$, pre-signature adaptability follows by the pre-signature adaptability of AS.
Extractability. We now reduce the extractability of AS' to the extractability of AS. Therefore, we assume by contradiction that an adversary $\mathcal{A}$ can break the extractability of AS' with non-negligible probability ε. We use $\mathcal{A}$ to build an algorithm $\mathcal{B}$ that breaks the extractability of AS as follows: $\mathcal{B}$ gets as input a public key vk and has access to a pSign, and a Sign, and a NewY oracle. $\mathcal{B}$ forwards vk to $\mathcal{A}$ and forwards the Sign, and NewY oracle requests of $\mathcal{A}$ to its own oracles. To answer pSign' oracle queries of $\mathcal{A}$ on a message m and a statement Y, $\mathcal{B}$ queries its pre-sign oracle twice on m and Y and concatenates the two pre-signatures. Eventually, $\mathcal{A}$ outputs a message-signature pair (m^*, σ^*) which $\mathcal{B}$ also outputs. If $\mathcal{A}$ wins extractability, then $\mathcal{B}$ also does, since σ^* extracts with a pre-signature $(\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ in AS' if and only if σ^* extracts with either $\widetilde{\sigma}_1$ or $\widetilde{\sigma}_2$. □

Lemma 24. *Let* Rel *be a relation with efficiently decidable statements. This means there exists an efficient algorithm that can check if* $Y \in \mathcal{L}_{\mathsf{Rel}}$. *Let* AS *be an adaptor signature scheme for* Rel *that is secure w.r.t. Definition 11 and achieves extractability. Then, the modified adaptor signature scheme* AS' *as for Fig. 6 is secure w.r.t. Definition 11.*

Proof. To prove Lemma 24, we show pre-signature correctness, pre-signature adaptability, and extractability. Since extractability implies unforgeability and witness extractability, AS' is secure w.r.t. Definition 11.
Pre-Signature Correctness. The pre-signature correctness of AS' follows directly from the pre-signature correctness of AS, since for all statements $(Y, y) \leftarrow \mathsf{RGen}(1^\lambda)$, all $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$, all $m \in \{0,1\}^{\ell_m}$, and all $\widetilde{\sigma}, \sigma \in \{0,1\}^*$: $\mathsf{pSign}'(\mathsf{sk}, m, Y) = \mathsf{pSign}(\mathsf{sk}, m, Y)$, $\mathsf{pVrfy}'(\mathsf{vk}, m, Y, \widetilde{\sigma}) = \mathsf{pVrfy}(\mathsf{vk}, m, Y, \widetilde{\sigma})$, $\mathsf{Adapt}'(\mathsf{vk}, \widetilde{\sigma}, y) = \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)$, and $\mathsf{Extract}'$ $(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y) = \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)$.
Pre-Signature Adaptability. The pre-signature adaptability of AS' follows by the pre-signature adaptability of AS, since pre-signature adaptability is only defined for $(Y, y) \leftarrow$

$\mathsf{RGen}(1^\lambda)$. So, pre-signature adaptability holds due to the same proof as for pre-signature correctness.

Extractability. We now reduce the extractability of AS' to the extractability of AS. Therefore, we assume by contradiction that an adversary $\mathcal{A}$ can break the extractability of AS' with non-negligible probability ε. We use $\mathcal{A}$ to build an algorithm $\mathcal{B}$ that breaks the extractability of AS as follows: $\mathcal{B}$ gets as input a public key vk and has access to a pSign, and a Sign, and a NewY oracle. $\mathcal{B}$ forwards vk to $\mathcal{A}$ and forwards the Sign, and NewY oracle requests of $\mathcal{A}$ to its own oracles. To answer pSign' oracle queries of $\mathcal{A}$ on a message m and a statement Y, $\mathcal{B}$ checks, if $Y \in \mathcal{L}_{\mathsf{Rel}}$. If so, it forwards the response of its pre-sign oracle on m and Y to $\mathcal{A}$. If not, it answers $\perp$ to $\mathcal{A}$. $\mathcal{B}$ simulates the extractability game perfectly for AS'. If $\mathcal{A}$ outputs its forgery (m^*, σ^*), $\mathcal{B}$ also outputs this. Since $\mathcal{T}_{\mathcal{B}} \subseteq \mathcal{T}_{\mathcal{A}}$, $\mathcal{B}$ wins the extractability game, whenever $\mathcal{A}$ does. □

Lemma 25. *Let* AS *be an adaptor signature scheme in the random oracle model that is secure w.r.t. Definition 11. Then, the modified adaptor signature scheme* AS' *as for Fig. 7 is secure in the random oracle model w.r.t. Definition 11.*

Proof. To prove Lemma 25, we show pre-signature correctness, pre-signature adaptability, unforgeability, and witness extractability.

Pre-Signature Correctness. The pre-signature correctness of AS' follows directly from the pre-signature correctness of AS, since pSign' outputs a pre-signature derived from pSign together with a random element. The algorithms pVrfy, Adapt, and $\mathsf{Extract}$ ignore the additional element.

Pre-Signature Adaptability. The pre-signature adaptability of AS' follows by the pre-signature adaptability of AS, since the algorithms pVrfy, Adapt, and $\mathsf{Extract}$ ignore the additional element.

Unforgeability. We now reduce the unforgeability of AS' to the unforgeability of AS. Therefore, we assume by contradiction that an adversary $\mathcal{A}$ can break the unforgeability of AS' with non-negligible probability ε. We use $\mathcal{A}$ to build an algorithm $\mathcal{B}$ that breaks the unforgeability of AS as follows: $\mathcal{B}$ gets as input a public key vk and has access to a pSign, and a Sign oracle. $\mathcal{B}$ forwards vk to $\mathcal{A}$ and forwards the Sign oracle requests of $\mathcal{A}$ to its own oracle. Then, $\mathcal{B}$ samples a uniform key sk'. To answer pSign' oracle queries of $\mathcal{A}$ on a message m and a statement Y, $\mathcal{B}$ queries a pre-signature $\widetilde{\sigma}$ on m and Y, and a signature σ on m from its own oracles. Then, $\mathcal{B}$ queries (sk', m) to the random oracle to obtain r_0 and computes $r_1 = \sigma \oplus r_0$. $\mathcal{B}$ flips a bit $b \in \{0, 1\}$ and forwards $(\widetilde{\sigma}, r_b)$ to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs a message m^*. $\mathcal{B}$ outputs the same message and receives a statement Y and a pre-signature $\widetilde{\sigma}$ on m^* and Y. $\mathcal{B}$ queries (sk', m^*) to its own random oracle to obtain r_0 and gives $(\widetilde{\sigma}, r_0)$ to $\mathcal{A}$. This is a perfect simulation of the unforgeability game to $\mathcal{A}$ since $\mathcal{A}$ can only see one pre-signature on m^*, and the probability that $\mathcal{A}$ guesses sk', or sk correctly is negligible. Hence, by the randomness of the RO, this is a perfect simulation, even if $\mathcal{B}$ does not embed a signature to the challenge pre-signature with probability $1/2$. The adversary $\mathcal{A}$ eventually outputs a signature forgery σ^*, which $\mathcal{B}$ also outputs. If $\mathcal{A}$ never queries the signing oracle on m^*, then $\mathcal{B}$ also does not, and thus $\mathcal{B}$ wins the unforgeability game whenever $\mathcal{A}$ does.

Witness Extractability. The witness extractability reduction is analogous to the unforgeability reduction. Again, the adversary is only allowed to see a single pre-signature on the challenge message, and the probability of guessing either sk' or sk correctly is negligible.

Hence, by the randomness of the RO, $\mathcal{B}$ wins the extractability game whenever $\mathcal{A}$ does so. $\square$

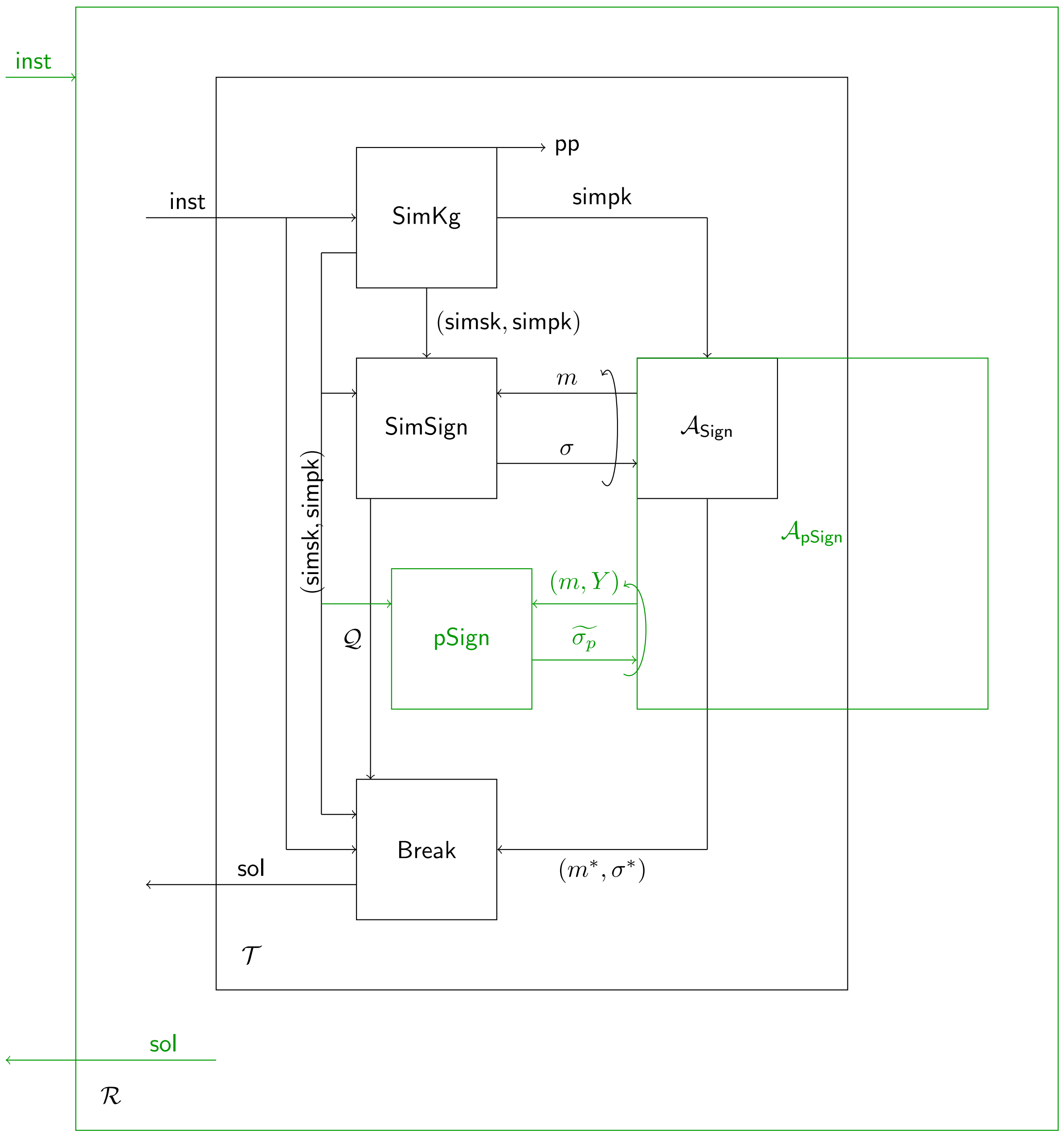

Figure 24: Using a Non-interactive Transparent Reduction to Answer Pre-signing Queries