Consecutive Adaptor Signature Scheme: From Two-Party to N-Party Settings

Abstract. Adaptor signatures have attracted attention as a tool to address scalability and interoperability issues in blockchain applications. Adaptor signatures can be constructed by extending common digital signature schemes that both authenticate a message and disclose a secret witness to a specific party. In Asiacrypt 2021, Aumayr et al. formulated the two-party adaptor signature as an independent cryptographic primitive. In this study, we extend their adaptor signature formulation to N parties, present its generic construction, and define the security to be satisfied. Then, we present a concrete construction based on Schnorr signatures and discuss the security properties.

1 Introduction

1.1 Background

An adaptor signature was first proposed by Andrew Poelstra et al. [29, 30] in 2017 as the concept of Scriptless Script and later formulated as an independent cryptographic primitive by Aumayr et al. [2]. Adaptor signatures have recently attracted attention as a tool to address issues such as scalability and interoperability for blockchain applications. An adaptor signature scheme is constituted as an extension of a digital signature scheme through a dialogue between two parties: a signer and a secretary. First, the signer generates a pre-signature based on a certain mathematical condition. The conditions are defined by a computationally hard algebraic relation between the public information and the secret information, such as the discrete logarithm problem or the preimage of a hash function. Next, the secretary, who possesses a secret witness for the above conditions, fits the pre-signature to create a valid adaptor signature. Once the valid adaptor signature is completed, the secret information is disclosed to the signer. A valid adaptor signature is a digital signature that is verifiable in the original signature scheme. Particularly in blockchain applications, a miner will not know that an ordinary signature is the output of an adaptor signature scheme and will simply verify it. At the same time, the two parties involved in generating the adaptor signature can embed conditions that are not restricted to the blockchain's scripting language. Thus, adaptor signatures can be used in off-chain payment instruments such as a payment channel network (PCN) [14, 2], which is an off-chain payment method, and an atomic swap [29, 17], which is a P2P transaction between different cryptocurrencies. Moreover, they can be used in scriptless blockchain applications. A PCN is a second-layer technology created to accelerate the transaction processing of cryptocurrencies [16].

To date, adaptor signatures have mainly been considered for applications like PCNs and atomic swaps, which are implemented via a dialoge between two parties. Accordingly, all existing studies on adaptor signature schemes have been based on this two-party case.

1.2 Related Works

As the adaptor signature was originally formulated as an independent cryptographic primitive by Aumayr et al., here, we mainly introduce related works that sought to analyze and improve adaptor signatures as cryptographic primitives.

Erwig et al. [13] proposed method for a general conversion method from IDs to adaptor signatures, following the work of Aumayr et al. Indeed, since that work, various adaptor signature schemes have been proposed as cryptographic primitives [14, 38, 22, 44]. The adaptor signature constructed by Aumayr et al. was based on Schnorr signature [33]. Lattice-based [14], isogeny-based [38], and code-based [22] signatures have been proposed as schemes that satisfy quantum security resistance. Along this line, Esgin et al. constructed a lattice-based adaptor signature (LAS) based on the Dilithium signature [11]. Tairi et al. constructed a isogeny-based adaptor signature by applying the Fiat-Shamir transformation [15] from the CSI-FiSh variant to the Schnorr type identification protocol [33, 38]. An optimized version (O-IAS) was then proposed by [38]. Klamti et al. proposed a code-based adaptor signature [22]. They used algebraic relations that were defined from the syndrome decoding problem to construct an adaptor signature based on Debris-Alazard et al.'s sign-based signature scheme of hash-and-sign [9].

On the security side, Erwig et al. [13] proposed an adaptor signature with re-randomizable keys to securely store secret information via algebraic relations with respect to the signing key. Dai et al. [8] strengthened the existing security definition, added a new security definition, and improved the security model of adaptor signatures. As for efficiency, Tu et al. [41] constructed an efficient adaptor signature based on ECDSA by generating zero-knowledge proofs in the pre-signature stage in a batch and offline.

As noted above, atomic swaps [10, 18, 41, 19] and payment channel networks (PCNs) [28, 39, 4, 38, 27, 3, 36, 24, 42, 26, 32] use adaptor signatures and have been actively studied in terms of various practical aspects. Both applications rely on technology to exchange secret information for signatures, which is precisely the functionality provided by adaptor signatures. Note again that adaptor signatures were originally designed to solve scalability and interoperability issues in blockchain applications, and they have various other applications [23, 34, 7]. Liu et al. [23] proposed a data sharing protocol on a blockchain, which is based on adaptor signatures and zero-knowledge proofs.

Regarding the functionality of adaptor signatures, Sui et al. [36] proposed a two-party, consecutive linkable ring adaptor signature (2P-CLRAS) to construct a PCN compatible with Monero [1], a privacy-preserving cryptocurrency. They constructed 2P-CLRAS from a sequential adaptor signature (CAS) by using a new cryptographic primitive called a Verifiable Consecutive One-Way Function

(VCOF). This led to the proposal of MoNet, a two-way, Monero-compatible PCN. Qin et al. [31] proposed a blind adaptor signature (BAS) based on blind signature schemes to construct a new privacy-preserving payment channel hub (PCH), BlindHub, and a privacy-preserving bidirectional PCN protocol, BlindChannel, in a PCH that supports off-chain payments between senders and receivers via an intermediary (called a tumbler). Hu and Chen [20] also proposed an anonymous, fair transaction scheme for electronic resources by using a new BAS technology. To reduce the PCN's computational complexity, Zhou et al. [43] proposed a new cryptographic primitive called a verifiable timed adaptor signature. Thvagarajan et al. [40] proposed a scheme that is similar to the adaptor signature, called a lockable signature, which does not require computationally difficult algebraic relations. Lockable signatures provide an effective signature scheme for constructing PCNs and can be seen as a special case of adaptor signatures. Finally, Ji et al. [21] proposed multi-adaptor signatures and threshold adaptor signatures based on Schnorr signature and Dilithium schemes, respectively. We discuss this work in more detail in Appendix A, as it is similar to our work in the context of multi signatures.

1.3 Our contribution

As explained above, while adaptor signatures are an important key technology for addressing issues such as the scalability of cryptocurrencies, simply repeating the current two-party setting is insufficient for ensuring security when consecutively generalizing adaptor signatures to multiple parties (i.e., an N-party protocol). In this paper, we introduce a novel concept, *the N-party consecutive adaptor signature* (or simply *N-party adaptor signature*), to address this problem. To accomplish this, we first propose a formal security model for three-party adaptor signatures and demonstrate a specific construction example using Schnorr signatures. Then, we rigorously establish that the proposed scheme precisely satisfies the defined security model. Following the discussion on three-party scenarios, we delve into the security and concrete constructions for N-party scenarios. The security proofs are demonstrated inductively by leveraging the established security among three parties.

Technical contributions. The paper's technical contributions include the generalization of constructing pre-signatures for pre-signatures. This allows the formation of concatenated adaptor signatures from two- to N-party settings. The mechanism for creating this "pre-signature of a pre-signature" is enabled by the additivity that appears in the syntax of adaptor signatures. We call our algorithm for this "pre-signature of a pre-signature" the $\mathsf{PreAdapt}$ algorithm. Furthermore, we provide rigorous security proofs. It is evident that if security holds for three-party adaptor signatures, then it easily extends to N-party settings. However, the security proof for three-party settings cannot be trivially derived from that of two-party settings, because of differences not only in the form of pre-signatures and the addition of a pre-adaptation oracle. but also the

Distinction from the 2-party setting. We explain the difference between our N-party adaptor signature and N iterations of a 2-party adaptor signature. A simple 2-party iteration is not sufficient for the security to be satisfied. For example, consider the scenario of an adaptor signature between Alice, Bob, and Charlie. After Alice and Bob execute the adaptor signature protocol, Bob transforms Alice's pre-signature and then creates another pre-signature for Charlie. If the two-party setup is repeated, then Alice and Bob, and Bob and Charlie, each independently generate adaptor signatures interactively. However, in this case Bob communicates with Charlie using messages from Alice after his interaction with Alice. If an attacker were to eavesdrop on the communication between Bob and Charlie, he would be able to observe Alice's information. Therefore, in order to create consecutive adaptor signatures between multiple parties, it is necessary to satisfy a higher security level than applying 2-party to each hop, i.e., being secure under an attacker who can observe the input and output of all users.

1.4 Application

In this section, we will consider one application example of the proposed N-party adaptor signature. As explained above, adaptor signatures are primarily used for applications that improve the efficiency of cryptocurrency transactions such as PCN and Atomic Swap. However, the functionality of adaptor signatures is not limited to a specific field, allowing for applications beyond cryptocurrency. For example, Liu et al. developed a secure data sharing protocol using adaptor signatures. We focus on the pre-signature compatibility of the adaptor signature, and consider the application of the proposed N-party adaptor signature to *a content provenance management system.* A content provenance management system is a technology that is attracting attention as a practical countermeasure to the problem of disinformation and misinformation caused by deep fakes, etc., which have become one of the social problems. The system creates a content provenance, including all of the histories in the supply chain of the content (e.g., shooting, editing, publishing), and attaches it to the content or records it on a blockchain. The content provenance can be verified by anyone, so that they can decide whether or not to trust the content they try to watch. C2PA (Coalition for Content Provenance and Authenticity) [5] is an international standardization body that standardizes technical specification [6] for the above system. In the C2PA specification, digital signatures are used to verify the content and its provenance information. However, the current C2PA specification does not take into account the relationship between entities. For instance, in the content creation field, it's common for a major production company (let's call it Company A) to subcontract certain tasks to another company (let's call it Company B), such as shooting or producing specific parts of the content of Company A. In such a case, using a standard digital signature compliant with C2PA, such as ECDSA, may record the provenance of the works as the sign of Company B, even if Company A has an originality and a copyright. Here, it is common for the contract to conceal the name of the subcontractor, Compny B, or not require it to be revealed. Therefore, once Company B sends pre-signed adaptor signatures along with the content they produce to Company A by employing adaptor signatures, Company A can then adapt Company B's pre-signed adaptor signatures into regular digital signatures. Based on the contract between Company A and B, Company A can maintain the provenance of the content as its own work while still allowing Company B to receive compensation. Moreover, in the content creation field, subcontracting often occurs hierarchically, with secondary and tertiary subcontracting being common. Therefore, by utilizing our proposed N-party adaptor signatures, it's possible to construct a content provenance management system in the supply chain among any number of entities without restrictions, while meeting the requirements mentioned above.

2 Preliminary for Adaptor Signatures

An adaptor signature scheme is essentially a two-step signing algorithm that is bound to a secret: a partial signature is first generated such that it can be completed only by a party knowing a certain secret, with the complete signature revealing that secret. More precisely, we define the adaptor signature scheme with respect to a digital signature scheme Σ and a hard relation R. For any statement $Y \in L_R$, a signer holding a secret key can produce a pre-signature w.r.t. Y on any message m. Such a pre-signature can be adapted into a valid signature on m if and only if the adaptor knows a witness for Y. Moreover, it must be possible to extract a witness for Y given the pre-signature and the adapted signature.

The adaptor signature scheme AS is constructed using a digital signature scheme $\Sigma = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ and a computationally intractable algebraic relation $(Y, y) \subseteq R$. Let $(Y, y) \leftarrow \mathsf{GenR}(\lambda)$ be a PPT algorithm that takes the security parameter λ as input and generates a pair comprising public and secret information related by an algebraic relation. For instance, when using the discrete logarithm problem, let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order q, as defined in the previous section. In this case, the computationally intractable algebraic relation R_g is defined as $R_g = \{(Y, y) | Y = g^y\} \subseteq \mathbb{G} \times \mathbb{Z}_q$. Other constructions, such as those based on lattice problems, may also exist, and the definition is tailored to the computational assumptions underlying the constructed adaptor signature's security.

In this paper, we use a signature scheme that satisfies $\mathsf{SUF\text{-}CMA}$ (strong existential unforgeability under chosen-message attack). $\mathsf{SUF\text{-}CMA}$ security guarantees that a PPT attacker with access to the signature oracle by entering his public key pk cannot generate a new valid signature for any message M.

3 Three-Party Adaptor Signatures

In this section, we describe a three-party adaptor signature scheme to prepare for our later N-party construction. The original adaptor signature scheme initially involves two entities, the secretary and the signer. However, in the proposed three-party scheme presented here, we consider three entities: U_1 as the secretary, U_2 as the main signer, and U_3 as a sub-signer. In this configuration, the sub-signer U_3 generates a pre-signature; the main signer U_2 generates another pre-signature for the same message, based on the pre-signature generated by U_3; and finally, the secretary U_1 performs adaptation to transform the pre-signature into a (normal) signature. At this point, it is evident that the adapted (normal) signature does not reveal the presence of U_2 or U_3, thus providing anonymity for the signers. We now define two primitives that serve as the foundation for constructing the adaptor signature. The first primitive is a digital signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$, where U_2 and U_3 possess pairs of public and private keys, denoted as $(\mathsf{pk}_2, \mathsf{sk}_2)$ and $(\mathsf{pk}_3, \mathsf{sk}_3)$, respectively. The signatures generated using these keys are represented as $\sigma_2 \leftarrow \mathsf{Sign}(\mathsf{pk}_2, \mathsf{sk}_2, M)$ and $\sigma_3 \leftarrow (\mathsf{pk}_3, \mathsf{sk}_3, M)$. Then, the second primitive is a hard relation R with statement/witness pairs (Y, y) (as defined at the beginning of Appendix B).

Syntax of three-party adaptor signatures. A three-party adaptor signature scheme w.r.t. a hard relation R and a signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ comprises six algorithms, $\Xi_{R,\Sigma}$=($\mathsf{PreSign}, \mathsf{PreVrfy}, \mathsf{PreAdapt}, \mathsf{Adapt}, \mathsf{PreExt}, \mathsf{Ext}$), with syntax defined as follows. $\hat{\sigma}_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M)$ is a PPT algorithm that takes as input a public key pk_3, a secret key sk_3, a statement $Y_2 \in R$, and a message $M \in \{0,1\}^*$, and outputs a pre-signature $\hat{\sigma}_3$. $b = \mathsf{PreVrfy}(Y_1, (\mathsf{pk}_2, \mathsf{pk}_3), (\hat{\sigma}_2), \hat{\sigma}_3), M)$ is a DPT algorithm that takes as input a statement $Y_1 \in R$, public keys $(\mathsf{pk}_2, \mathsf{pk}_3)$, pre-signatures $(\hat{\sigma}_2, \hat{\sigma}_3)$, and a message M, and outputs a bit b. $\mathsf{PreAdapt}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3, M)$ is a PPT algorithm[1] that takes as input a pair of hard relations, (Y_2, y_2), a statement Y_1, a public key pk_3, a pair of keys $(\mathsf{sk}_2, \mathsf{pk}_2)$, a pre-signature $\hat{\sigma}_3$, and a message M; then, it outputs a pre-signature $\hat{\sigma}_2$. $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \hat{\sigma}_2, M)$ is a DPT algorithm that takes as input a pair of a statement and a witness, (Y_1, y_1), a public key pk_2, a pre-signature $(\hat{\sigma})$, and a message M. $\mathsf{PreExt}(Y_2, \hat{\sigma}_3, \hat{\sigma}_2)$ is a DPT algorithm that takes as input a public statement Y_2 and pre-signatures $(\hat{\sigma}_3, \hat{\sigma}_2)$, and outputs either a witness y_2' such that $(Y_2, y_2') \in R$, or $\perp$. $\mathsf{Ext}(Y_1, \hat{\sigma_2}, \sigma_2)$ is a DPT algorithm that takes as input a public statement Y_1, a pre-signature $\hat{\sigma}_2$, and an (original) signature σ_2, and outputs either a witness y_1' such that $(Y_1, y_1') \in R$, or $\perp$.

3.1 Concrete Construction of Three-Party Adaptor Signatures

In this section, we extend the two-party adaptor signature defined in Section 2 to describe a specific instantiation of the Schnorr-based three-party adaptor signature scheme outlined in Fig. 1. For Schnorr signatures Σ_{Sch} and a hard relation $R_g := \{(Y, y) | Y = g^y\}$, we show the concrete construction of an N-party

[1] The $\mathsf{PreAdapt}$ algorithm simultaneously performs internal processing of the Adapt and $\mathsf{PreSign}$ algorithms. While it includes elements of a DPT algorithm because of this simultaneous processing, the overall procedure involves probabilistic steps when generating pre-signatures. Therefore, it is defined as a PPT algorithm.

adaptor signature scheme $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$ for the case of $N = 3$. Here, $\mathcal{H}(\cdot)$ denotes any cryptographic hash function, and $\mathbb{Z}_q^*$ denotes the set of all integers from 1 to q, excluding 0. First, we denote the three entities in this scheme as U_1, U_2, and U_3. In the construction of the three-party adaptor signature, an algorithm's subscript (e.g., $\mathsf{PreSign}_{U_3}$) corresponds to the entity executing the algorithm, and a subscript in an argument (e.g., Y_2 in $\mathsf{PreSign}_{U_3}$ or 3 in $\hat{\sigma}_3$) corresponds to the entity that initially owns (or generates) that value.

Fig. 1. Concrete construction: Schnorr-based three-party adaptor signatures.

```
$U_1$: $\underline{(Y_1, y_1) \leftarrow \mathsf{GenR}(\lambda);}$
    $y_1 \leftarrow \mathbb{Z}_q^*$, $Y_1 := g^{y_1}$,
    return $Y_1$ to $U_2$.

$U_2$: $\underline{(\mathsf{sk}_2, \mathsf{pk}_2) \leftarrow \mathsf{KeyGen}(\lambda),}$
       $\underline{(Y_2, y_2) \leftarrow \mathsf{GenR}(\lambda);}$
    $\mathsf{sk}_2 := x_2 \leftarrow \mathbb{Z}_q$, $\mathsf{pk}_2 = X_2 := g^{x_2} \in \mathbf{G}$,
    $y_2 \leftarrow \mathbb{Z}_q^*$, $Y_2 := g^{y_2}$,
    return $Y_2$ to $U_3$ and $\mathsf{pk}_2$ to $U_1, U_3$.

$U_3$: $\underline{(\mathsf{sk}_3, \mathsf{pk}_3) \leftarrow \mathsf{KGen}(\lambda);}$
    $\mathsf{sk}_3 := x_3 \leftarrow \mathbb{Z}_q$, $\mathsf{pk}_3 = X_3 := g^{x_3} \in \mathbf{G}$,
    return $\mathsf{pk}_3$ to $U_1$, $U_2$.

$U_3$: $\underline{\hat{\sigma}_3 \leftarrow \mathsf{PreSign}_{U_3}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M);}$
    $k \leftarrow \mathbb{Z}_q$, $r_3 := \mathcal{H}(X_3 || g^k Y_2 || M)$,
    $s_3 := k + r_3 \cdot x_3$, $\hat{\sigma}_3 := (r_3, s_3)$,
    return $(\hat{\sigma}_3, M)$ to $U_1$, $U_2$.

$U_2$: $\underline{0/1 = \mathsf{PreVrfy}_{U_2}(Y_2, \mathsf{pk}_3, \hat{\sigma}_3, M);}$
    return 1
    if $r_3 = \mathcal{H}(X_3 || g^{s_3} \cdot X_3^{-r_3} \cdot Y_2 || M)$.

$U_2$: $\underline{\hat{\sigma}_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3, M);}$
    $k' \leftarrow \mathbb{Z}_q$, $r_2 := \mathcal{H}\left(X_2 || g^{k'} Y_1 || M\right)$,
    $s_2 := k' + r_2 \cdot x_2$, $s_3' := s_3 + y_2$,
    $\hat{\sigma}_2 = (r_2, s_2, s_3')$,
    return $(\hat{\sigma}_2, \hat{\sigma}_3, M)$ to $U_1$, and $\hat{\sigma}_2$ to $U_3$.

$U_1$:
$\underline{b = \mathsf{PreVrfy}_{U_1}(Y_1, Y_2, \mathsf{pk}_2, \mathsf{pk}_3, \hat{\sigma}_2, \hat{\sigma}_3, M);}$
    return 1 if $r_3 = \mathcal{H}(X_3 || g^{s_3} X_3^{-r_3} Y_2 || M)$
    and $r_2 = \mathcal{H}(X_2 || g^{s_2} X_2^{-r_2} Y_1 || M)$.

$U_1$: $\underline{\sigma_2 = \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \hat{\sigma}_2, M);}$
    $s_1 := s_2 + y_1$, $\sigma_2 = (r_2, s_1)$,
    return $\sigma_2$ to $U_2$.

$U_2$: $\underline{y_1' / \perp = \mathsf{Ext}_{U_2}(Y_1, (\hat{\sigma}_2, \sigma_2));}$
    $y_1' := s_1 - s_2$,
    return $y_1'$ if $(Y_1, y_1') \in R$, otherwise,
    return $\perp$.

$U_3$: $\underline{y_2' / \perp = \mathsf{PreExt}_{U_3}(Y_2, \hat{\sigma}_3, \hat{\sigma}_2);}$
    $y_2' := s_3' - s_3$,
    return $y_2'$ if $(Y_2, y_2') \in R$, otherwise,
    return $\perp$.
```

3.2 Security Definitions for Three-Party Adaptor Signature Scheme

We now define the security definitions from two-party adaptor signature scheme. In total, four security requirements must be satisfied, the most important and core of which is Unforgeability. In this paper, we focus on Unforgeability, while the other security properties are described in the Appendix. Specifically, given a theorem indicating that the proposed scheme is secure, we present each of the four security properties as a lemma. Here, we show one of the lemmas for Unforgeability. Existential unforgeability under chosen message attack in the context of three-party adaptor signatures ($\mathsf{3\text{-}aEUF\text{-}CMA}$) is an extension of the unforgeability (Definition 6 in Appendix C.2) for adaptor signatures to the three-party setting. In the three-party case, because the content of signatures depends on which entity generates them, we need to consider unforgeability for two separate dialogues involving all three entities, as classified into two cases. First, for unforgeability between entities U_3 and U_2, U_3 generates pre-signatures via the $\mathsf{PreSign}$ algorithm, and the attacker attempts to forge signatures via the signature/pre-signature oracle. Second, for unforgeability between U_1 and U_2, U_2 generates pre-signatures via the $\mathsf{PreAdapt}$ algorithm, and the attacker tries to forge signatures via the signature/pre-adaptation oracle. We define $\mathsf{3\text{-}aEUF\text{-}CMA}$ as follows:

Definition 1 (Existential unforgeability for three parties). *A three-party adaptor signature scheme* $\mathsf{3\text{-}AS}_{R,\Sigma}$ *is 3-aEUF-CMA secure if for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\Pr[\mathsf{3\text{-}aSigForge}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R,\Sigma}}(\lambda) = 1] + \Pr[\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{3\text{-}AS}_{R,\Sigma}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

where the experiments $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R,\Sigma}}$ *and* $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{3\text{-}AS}_{R,\Sigma}}$ *are as defined in the figure Table 1.*

3.3 Proof of Unforgeability for Three-Party Adaptor Signature Scheme

In this section, we show the security of three-way adaptor signatures. An adaptor signature is secure if the following four conditions hold: pre-signature correctness, pre-signature adaptability, existential unforgeability, and witness extractability. Therefore, the security theorem 1 is proved from the Lemma showing these four. Among them, the lemma for pre-signature correctness and pre-signauture adaptability can be verified by computation, and witness extractability can be verified by a small change and a device for the system of existential unforgeability In this paper, we mainly focus on the intrinsic unforgeability, and the other proofs are presented in the Appendix.

Theorem 1. *If Schnorr signature scheme* Σ_{Sch} *is SUF-CMA-secure, and* R_g *is a computationally hard algebraic relation, then* $\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$ *in Fig. 1 is secure in the ROM.*

Table 1. Experiments $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R,\Sigma}}$ and $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{3\text{-}AS}_{R,\Sigma}}$

```
$\mathsf{3\text{-}aSigForge}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$
2 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
3 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
4 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
5 : $\sigma_3^* \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
6 : $\sigma_2^* \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
7 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1, Y_2)$
8 : return $(M^* \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2^*, M^*))$
```

```
$\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{3\text{-}AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$
2 : $(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
3 : $(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
4 : $M^* \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk}_3)$
5 : $\sigma_3^* \leftarrow \mathsf{PreSign}_{U_3}((\mathsf{pk}_3, \mathsf{sk}_3), Y_3, M^*)$
6 : $\sigma_{n-1} \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\sigma_3^*, Y_2)$
7 : return $(M^* \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

```
$\mathcal{O}_{pA}(M, (Y_1, Y_2), \sigma_3)$
1 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M)$
2 : $Q := Q \cup \{M\}$
3 : return $\sigma_2$
```

```
$\mathcal{O}_S^{A_i}(M)$
1 : $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, M)$
2 : $Q := Q \cup \{M\}$
3 : return $\sigma_i$
```

```
$\mathcal{O}_{pS}(M, Y_2)$
1 : $\sigma_3 \leftarrow \mathsf{PreSign}(\mathsf{sk}_3, Y_2, M)$
2 : $Q := Q \cup \{M\}$
3 : return $\sigma_3$
```

Lemma 1 (3-aEUF-CMA security.). *Assuming that Schnorr digital signature scheme Σ_{Sch} is SUF-CMA secure and R_g is a hard relation, the three-party adaptor signature scheme $\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}$, as defined in Fig.1, is 3-aEUF-CMA secure.*

Before formally proving this lemma, we discuss the main idea behind the proof intuitively. The goal is to reduce the forgery resistance of the three-party adaptor signature scheme to the strong resistance of the standard Schnorr scheme. In the three-party scheme, we have two cases to consider: an adversary A_1 between U_1 and U_2, or an adversary A_2 between U_2 and U_3, where the adversary wins the 3-aSigForge experiment as a PPT attacker. We design an adversary (or simulator) S for this purpose. First, case (i) can be proved by using almost the same game-hopping steps as in the proof of the two-party adaptor signature scheme [Lemma 4 (aSigForge) in [2]], so we skip the proof here. Next, for case (ii), the proof follows a completely different approach from the two-party case. The technical challenge is that A_1 can access not only the interaction between U_1 and U_2 but also the information exchanged between U_2 and U_3 that A_2 had. Specifically, A_2 has access to previous pre-signatures.

Proof of Lemma 1. We consider two cases, (i) and (ii) as explained above, and we perform several game hops in each case to prove Lemma 1. For case (i), we follow a similar procedure to the proof of unforgeability in the original two-party scenario of Lemma 4 in [2]. Then, for case (i), we demonstrate each game hop and reduction loss via a sketch proof.

For case (i), we have the following game definitions for $\mathsf{strongSigForge}$ and G_0 to G_4.

Game G_0: The original $\mathsf{3\text{-}aEUF\text{-}CMA}$ game, $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{AS}_{R,\Sigma}}$.
Game G_1: An abort game for when the adversary forges a pre-signature for the challenge public statement Y^* without knowing the secret witness s^*.
Game G_2: An abort game for when queries to the oracle overlap.
Game G_3: A game where the pre-signature oracle returns a regular signature.
Game G_4: A game where the pre-signature given to the adversary is turned into a regular signature.
Game $\mathsf{strongSigForge}$: A $\mathsf{SUF\text{-}CMA}$ game for regular signatures.

The reduction loss for the above is as follows, and it can be directly obtained from the original two-party scenario:

$$\begin{aligned}
&\Pr[\mathsf{3\text{-}aSigForge}_{A_2,\mathsf{3\text{-}AS}}(\lambda)=1]\\
&=\Pr[G_0=1]\\
&\le \Pr[G_1=1]+v_1(\lambda)\\
&\le \Pr[G_2=1]+v_1(\lambda)+v_2(\lambda)\\
&=\Pr[G_3=1]+v_1(\lambda)+v_2(\lambda)\\
&\le \Pr[G_4=1]+v_1(\lambda)+v_2(\lambda)\\
&=\Pr[\mathsf{strongSigForge}_{S^{A_2},\mathsf{3\text{-}AS}}(\lambda)=1]+v_1(\lambda)+v_2(\lambda),
\end{aligned} \tag{1}$$

where v_1 and v_2 are the negligible functions in λ.

Next, for case (ii), we have $\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{AS}_{R,\Sigma}}$.

Game $\mathbf{G}_0$: This game, which is formally defined in Table 5, corresponds to the original $\mathsf{3\text{-}aSigForge}$, where the adversary A_1 has to produce a valid forgery for a message m of his choice, while having access to a pre-signature oracle $\mathcal{O}_{\mathrm{pS}}$, a signature oracle $\mathcal{O}_{\mathrm{S}}$, and a pre-adaptation oracle $\mathcal{O}_{\mathrm{pA}}$. Since we are in the ROM, the adversary (as well as all of the scheme's algorithms) also has access to a random oracle $\mathcal{H}$. The simulator S wins if $b=1$ and $M^* \notin Q$.

$$\Pr[\mathsf{3\text{-}aSigForge}_{\mathcal{A}_2,\mathsf{AS}_{R,\Sigma}}(\lambda)=1]=\Pr[\mathbf{G}_0=1]. \tag{2}$$

Game $\mathbf{G}_1$: This game, which is formally defined in Table 6, works exactly like $\mathbf{G}_0$ with the following exception. When the adversary outputs a forgery σ_1^*, the game $\mathbf{G}_1$ checks whether completion of the pre-signature $\hat{\sigma}_2$ by using the secret value y_1 yields σ_1^*. If so, the game aborts. The difference between $\mathbf{G}_0$ and $\mathbf{G}_1$ is recorded in Table 6.

Claim of Game $\mathbf{G}_1$. Let Bad_1 be the event that $\mathbf{G}_1$ aborts. Then, $\Pr[\mathsf{Bad}_1] \le v_1(\lambda)$, where v_1 is a negligible function in λ.

Since games $\mathbf{G}_1$ and $\mathbf{G}_0$ are equivalent except if event Bad_1 occurs, it holds that

$$\Pr[\mathbf{G}_0 = 1] \le \Pr[\mathbf{G}_1 = 1] + v_1(\lambda), \tag{3}$$

where v_1 means the probability of breaking the hard relation R.

Game $\mathbf{G}_2$: This game, which is formally defined in Table 7, behaves similarly to the previous game, with the only difference being in the $\mathcal{O}_{\mathrm{pS}}$ oracle. In this game, the $\mathcal{O}_{\mathrm{pS}}$ oracle first makes a copy of the list H before executing the algorithm $\mathsf{PreSign}$. Then, it extracts the randomness used during the $\mathsf{PreSign}$ algorithm, and checks whether, before the signing algorithm's execution, a query of the form $\mathsf{pk}_3||K||M$ or $\mathsf{pk}_3||K \cdot Y_2||M$ was made to $\mathcal{H}$ by checking whether $H'[\mathsf{pk}_3||K||M] \neq \perp$ or $H'[\mathsf{pk}_3||K \cdot Y_2||M] \neq \perp$. If such a query was made, the game aborts. The difference between $\mathbf{G}_1$ and $\mathbf{G}_2$ is recorded in Table 7.

Claim of Game $\mathbf{G}_2$. Let Bad_2 be the event that $\mathbf{G}_2$ aborts in $\mathcal{O}_{pS}$. Then $\Pr[\mathsf{Bad}_2] \le v_2(\lambda)$, where v_2 is a negligible function in n.

Game $\mathbf{G}_3$: This game, which is formally defined in Table 8, behaves similarly to the previous game, but with several differences in the oracle $\mathcal{O}_{\mathrm{pA}}$. In this game, $\mathcal{O}_{\mathrm{pA}}$ first makes a copy of the list H before executing $\mathsf{PreAdapt}$. Afterward, it extracts the randomness values r_2, s_2, and s'_3 that were used in the $\mathsf{PreAdapt}$ algorithm. For r_2 and s_2, it checks whether, before the signing algorithm's execution, a query of the form $\mathsf{pk}_2||K_2||M$ or $\mathsf{pk}_2||K_2 \cdot Y_1||M$ was made to $\mathcal{H}$ by checking whether $H'[\mathsf{pk}_2||K_2||M] \neq \perp$ or $H'[\mathsf{pk}_2||K_2 \cdot Y_1||M] \neq \perp$. If such a query was made, the game aborts. The difference between $\mathbf{G}_2$ and $\mathbf{G}_3$ is recorded in Table 8.

Regarding s'_3, it is used when extracting the witness y'_2 ($y'_2 := s'_3 - s_3$). The adversary checks whether its forged witness y'_2 corresponds (by chance) to the challenge statement Y^*_2 on the oracle side. (The same verification performed by the simulator in $\mathbf{G_1}$ is performed here on the oracle side.) In other words, on the side of the pre-adaptation oracle $\mathcal{O}_{pA}$, if y'_2 is computed from $\sigma'_3 = (r_3, s_3)$ and $\sigma_2 = (r_2, s_2, s'_3)$ as $y'_2 = s'_3 - s_3$, and if $(Y^*_2, y'_2) \in R^*$, then the game aborts. While it aborts if a valid witness y'_2 for the challenge statement Y^*_2 exists, a separate list S is prepared because of possible overlapping queries.

If A_1 uses a challenge M^* as an oracle query, this can be determined by checking whether $M \notin \mathcal{Q}$; similarly, if A_1 uses $M(\neq M^*)$ and a challenge Y^*_1 as oracle queries, this can be determined by the Adapt algorithm in line 39 of $\mathbf{G}_3$. For the case where A_1 uses $M(\neq M^*)$ and a challenge Y^*_2, however, further consideration is needed. Therefore, on the simulator side, S executes the algorithm $y'_1 \leftarrow \mathsf{Ext}(Y_1, \hat{\sigma}_2, \sigma_2)$. If the forged σ^*_2 output by A_1 corresponds to the legitimate witness y^*_1 derived from the challenge Y^*_1 when $\sigma'_2 \leftarrow \mathsf{PreAdapt}$ is computed using $M(\neq M^*)$ and the challenge Y^*_2, then the game aborts. This probability is bounded by the probability of breaking the hard relation, at most.

Claim of Game $\mathbf{G}_3$*.* Let Bad_3 be the event that $\mathbf{G}_3$ aborts in $\mathcal{O}_{pA}$. Then $\Pr[\mathsf{Bad}_3] \leq v_3(\lambda)$, where v_3 is a negligible function in n.

Game $\mathbf{G}_4$: In this game, which is formally defined in Table 9 upon an $\mathcal{O}_{pS}$ query, the game produces a valid full signature $\tilde{\sigma} = (r, s) = (\mathcal{H}(pk||K||m), k + rs \cdot k)$ and adjusts the global list H as follows. It assigns the value stored at position $pk||K||m$ to $H[pk||K \cdot Y||m]$ and samples a fresh random value for $H[pk||K||m]$. These changes make the full signature $\tilde{\sigma}$ "look like" a pre-signature to the adversary A, because it obtains the value $H[pk||K||m]$ upon querying the random oracle on $pk||K \cdot Y||m$. The adversary can only notice the changes in this game if the random oracle was previously queried on either $pk||K||m$ or $pk||K \cdot Y||m$. This case is captured in the previous game, and it thus holds that $\Pr[G_3 = 1] = \Pr[G_4 = 1]$.

Game $\mathbf{G_5}$: This game, which is formally defined in Table 10, aims to appear like a pre-signature to the adversary upon an $\mathcal{O}_{pA}$ query. However, because a pre-signature's structure differs from that of a regular signature, adversaries may notice the changes in this game, unlike with $\mathbf{G_4}$. Therefore, to simulate $\hat{\sigma}_2 = (r_2, s_2, s'_3)$, where the first two components are indistinguishable from a regular signature $\sigma_2 = (r_2, s_1)$ and s'_3 is just random noise, we need to perform a similar procedure to that in $\mathbf{G_4}$, ensuring that $\sigma_2 = (r_2, s_1)$ remains indistinguishable and replacing s'_3 with a random value. The difference between $\mathbf{G}_4$ and $\mathbf{G}_5$ is recorded in Table 10. When using a randomly chosen s'_3 to extract y'_2, the probability that y'_2 corresponds to the challenge Y^*_2 is bounded by the advantage of breaking the hard relation, denoted as v_1. Hence, $\Pr[G_4 = 1] = \Pr[G_5 = 1] + v_1(\lambda)$ holds.

Game $\mathbf{G}_6$: In this game, which is formally defined in Table 11, the pre-signature generated upon A outputting the message M is created by modifying a full signature to a pre-signature.

The simulator S modifies the pre-signatures passed to adversary A_1 to pre-signatures converted from regular signatures. Specifically, from regular signatures σ'_2 and σ'_3, S creates $\sigma_2 = \text{Adapt}(\sigma'_2, -y_1)$ and $\sigma_3 = \text{Adapt}(\sigma'_3, -y_1)$. The difference in this transformation lies in k_2 and k_3 becoming $k'_2 = k_2 - y_1$ and $k'_3 = k_3 - y_2$; however, because k is uniformly random, it is indistinguishable, and A_1 cannot determine whether a signature is a pre-signature or a regular signature. This transformation can be viewed as k being modified to $k' = k - y_2$. Because k is chosen uniformly at random, and because, according to Proposition 1, the adversary's view is identical between this game and previous game, it holds that $\Pr[G_5 = 1] = \Pr[G_6 = 1]$.

Game strongSigForge: In analogous fashion, we seek to establish the existence of a simulator that can faithfully reproduce G_6 while harnessing the capabilities of A_1 to achieve success in the **strongSigForge** game. Here, we succinctly delineate how the simulator responds to oracle queries. For a comprehensive formal exposition of the simulator, refer to Table 12.

Signature queries: When the adversary A_1 queries oracle $\mathcal{O}_S$ with an input M, the simulator S forwards M to oracle $\mathcal{O}^{Sch}_{\text{Sign}}$ and relays the response to A_1.

Random oracle queries: When A_1 queries oracle $\mathcal{H}$ with input x, if $\mathcal{H}[x] =\perp$, then S queries $\mathcal{H}^{Sch}(x)$; otherwise, it returns $\mathcal{H}[x]$.

Pre-signature queries: 1. When A_1 queries oracle $\mathcal{O}_{pS}$ with input (M, Y_2), S forwards M to oracle $\mathcal{O}^{Sch}_{\text{Sign}}$ and receives a signature $\sigma_3 = (r_3, s_3)$ $(r_3 = \mathcal{H}^{Sch}(pk_3||K_3||M))$. 2. If oracle $\mathcal{H}$ was previously queried on $(pk_3||K_3||M)$ or $(pk_3||K_3 \cdot Y_2||\text{"}M\text{"})$, S aborts. 3. S programs the random oracle $\mathcal{H}$ such that queries made by A_1 on input $(pk_3||K_3 \cdot Y_2||\text{"}M\text{"})$ are answered with the value $\mathcal{H}^{Sch}(pk_3||K_3||M)$, and queries on input $(pk_3||K_3||M)$ are answered with $\mathcal{H}^{Sch}(pk_3||K_3 \cdot Y_2||M)$. 4. S returns σ_3 to A_1.

Pre-Adaptation queries: 1. When A_1 queries oracle $\mathcal{O}_{pA}$ with input (M, Y_1, Y_2, σ_2), S forwards M to oracle $\mathcal{O}^{Sch}_{\text{Sign}}$ and receives a signature $\sigma_2 = (r_2, s_2)$ $(r_2 = \mathcal{H}^{Sch}(pk_2||K_2||M))$. 2. If oracle $\mathcal{H}$ was previously queried on $(pk_2||K_2||M)$ or $(pk_2||K_2 \cdot Y_1||\text{"}M\text{"})$, S aborts. 3. S programs the random oracle $\mathcal{H}$ such that queries made by A_1 on input $(pk_2||K_2 \cdot Y_1||\text{"}M\text{"})$ are answered with the value $\mathcal{H}^{Sch}(pk_2||K_2||M)$, and queries on input $(pk_2||K_2||M)$ are answered with $\mathcal{H}^{Sch}(pk_2||K_2 \cdot Y_1||M)$. 4. S returns σ_2 to A_2.

Challenge Phase: S selects $(Y_1, y_1)(Y_2, y_2) \leftarrow GenR(1^n)$ and runs A_1 on pk_2, pk_3 and Y_1, Y_2. If A_1 outputs a challenge message M^*, then S queries the Sign^{Sch} oracle with input M^*. If A_1 outputs a forged signature σ^*, then S outputs (M^*, σ^*) as its own forgery.

The main difference between the simulation and G_6 lies in the syntax. Instead of generating public and secret keys and calculating the algorithm Sign_{sk} and random oracle $\mathcal{H}$, the simulator S uses its own oracles Sign^{Sch} and $\mathcal{H}^{Sch}$. Thus, S perfectly simulates G_6. We still need to demonstrate that S can use the forgery output by A_1 to win the **strongSigForge** game.

Claim of Game **strongSigForge***.* (M^*, σ^*) is a valid forgery of **strongSigForge**.

As S provides a perfect simulation of G_6, from games $\mathbf{G_0}$ to $\mathbf{G_6}$, we have the following:

$$\begin{aligned}
&\Pr[\textsf{3-aSigForge}_{\mathcal{A}_2,\textsf{AS}_{R_g,\Sigma}}(\lambda) = 1] \qquad (4)\\
&= \Pr[\mathbf{G}_0 = 1] \leq \Pr[G_1 = 1] + v_1(\lambda)\\
&\leq \Pr[G_2 = 1] + v_1(\lambda) + v_2'(\lambda)\\
&= \Pr[G_3 = 1] + 2v_1(\lambda) + v_2'(\lambda) + v_3'(\lambda)\\
&= \Pr[G_4 = 1] + 2v_1(\lambda) + v_2'(\lambda) + v_3'(\lambda)\\
&= \Pr[G_5 = 1] + 3v_1(\lambda) + v_2'(\lambda) + v_3'(\lambda)\\
&= \Pr[G_6 = 1] + 3v_1(\lambda) + v_2'(\lambda) + v_3'(\lambda)\\
&= \Pr[\textsf{strongSigForge}_{S^{A_2},\textsf{3-AS}}(\lambda) = 1] + 3v_1(\lambda) + v_2'(\lambda) + v_3'(\lambda).
\end{aligned}$$

Then, from Equations (1) and (4), the overall reduction loss for three-party unforgeability is as follows:

$$\Pr[\textsf{3-aSigForge}_{\mathcal{A}_1,\textsf{AS}_{R_g,\Sigma}}(\lambda) = 1] + \Pr[\textsf{3-aSigForge}_{\mathcal{A}_2,\textsf{AS}_{R_g,\Sigma}}(\lambda) = 1] \leq \textsf{negl}(\lambda).$$

□

4 N-Party Adaptor Signatures

In this section, we extend the two-party adaptor signatures defined in Section 2 to construct an N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R,\Sigma}$. Here, we denote the N entities as $U_1, \cdots, U_i, \cdots, U_n$. Each entity is classified into one of three types: U_n as the entity that generates the initial pre-signature, U_1 as the entity that finally adapts from pre-signatures to a regular signature, and U_i as all the other entities. In constructing N-party adaptor signatures, we assume the same digital signature scheme and computationally hard algebraic relation used in the two-party case. Additionally, the subscripts for each algorithm (e.g., U_n in $\mathsf{PreSign}_{U_n}$) correspond to the entity executing the algorithm, and the subscripts for each argument (e.g., i in Y_i or n in σ_n) correspond to the entity that initially owns (or generates) that value.

Syntax of N-Party Adaptor Signatures Setup. U_1 executes the algebraic relation generation algorithm $(Y_1, y_1) \leftarrow \mathsf{GenR}(1^n)$; the U_i $(2 \leq i < n)$ execute the key generation algorithm $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\lambda)$ and the algebraic relation generation algorithm $(Y_i, y_i) \leftarrow \mathsf{GenR}(1^n)$; and U_n executes the key generation algorithm $(sk_n, pk_n) \leftarrow \mathsf{KeyGen}(\lambda)$.
Pre-signing: $\sigma_n \leftarrow \mathsf{PreSign}_{U_n}((\mathsf{pk}_n, \mathsf{sk}_n), Y_{n-1}, M)$**.** The pre-signing algorithm $\mathsf{PreSign}_{U_n}$ is executed by U_n and takes as input the key pair $(\mathsf{pk}_n, \mathsf{sk}_n)$ of U_n, the public information Y_{n-1} of U_{n-1}, and the message M; then, it outputs the pre-signature σ_n. Note that all entities except U_1 generate pre-signatures, but because entities other than U_1 and U_n generate pre-signatures with the $\mathsf{PreAdapt}$ algorithm, only U_n executes this pre-signing algorithm.
Pre-verification: $0/1 \leftarrow \mathsf{PreVrfy}_{U_i}(\{Y_j\}_{j=i}^{n-1}, \{\mathsf{pk}_j\}_{j=i+1}^{n}, \{\sigma_j\}_{j=i+1}^{n}, M)$**.** The pre-verification algorithm $\mathsf{PreVrfy}_{U_i}$ is executed by the U_i $(1 \leq i < n)$. It takes as input the public information $(Y_i, \ldots, Y_{n-1})$ from U_i to U_{n-1}, the pre-signatures $(\sigma_{i+1}, \ldots, \sigma_n)$ and public keys $(\mathsf{pk}_{i+1}, \ldots, \mathsf{pk}_n)$ generated by U_{i+1} to U_n, and the message M, and it performs pre-signature verification. It outputs 1 if the signature is accepted, or 0 otherwise.
Pre-adaptation: $\sigma_i \leftarrow \mathsf{PreAdapt}_{U_i}((Y_i, y_i), Y_{i-1}, \mathsf{pk}_{i+1}, (\mathsf{sk}_i, \mathsf{pk}_i), \sigma_{i+1}, M)$**.** The pre-adaptation algorithm $\mathsf{PreAdapt}_{U_i}$ is executed by the U_i $(2 \leq i < n)$. It takes as input the algebraic relation pair (Y_i, y_i) of U_i, the public information Y_{i-1} of U_{i-1}, the public key pk_{i+1} of U_{i+1}, the key pair $(\mathsf{sk}_i, \mathsf{pk}_i)$ of U_i, the pre-signature σ_{i+1} generated by U_{i+1}, and the message M; then, it outputs the pre-signature σ_i.
Adaptation: $\sigma_1 \leftarrow \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M)$: The adaptation algorithm Adapt_{U_1} is executed by U_1. It takes as input the algebraic relation pair (Y_1, y_1) of U_1, the public key pk_2 of U_2, the signature σ_2 generated by U_2, and the message M, and it outputs the signature σ_1.
Extraction: $y_{i-1} \leftarrow \mathsf{Ext}_{U_i}(Y_{i-1}, \sigma_i, \sigma_{i-1})$: The extraction algorithm Ext_{U_i} is executed by the U_i $(2 \leq i \leq n)$. It takes as input the public information Y_{i-1}

of U_{i-1} and the signatures σ_i and σ_{i-1} generated by U_i and U_{i-1}, respectively, and it outputs the secret information y_{i-1}.

4.1 Security of N-party Adaptor Signature Scheme

We now extend the security definitions of Section 3.2 to define the security properties that the N-party adaptor signatures, as defined in Section 4, should satisfy.

Existential unforgeability against chosen-plaintext attacks for N-party adaptor signatures ($\mathsf{N\text{-}aEUF\text{-}CMA}$) extends the unforgeability definition for adaptor signatures (Definition 6) to N parties. Here, because the signature content depends on the generating entity, it is necessary to consider unforgeability for n entities with $n-1$ interactions each. Hence, we consider two cases. The first case is unforgeability between U_n and U_{n-1}. In this case, U_n generates a pre-signature via the $\mathsf{PreSign}$ algorithm, and the adversary attempts to forge signatures via the signature/pre-signature oracle. The second case is unforgeability between U_i and U_{i+1} for $1 \leq i \leq n-2$. Here, U_{i+1} generates a pre-signature via the $\mathsf{PreAdapt}$ algorithm, and the adversary attempts to forge signatures via the signature/pre-adaptation oracle. We define $\mathsf{N\text{-}aEUF\text{-}CMA}$ as follows.

Definition 2 (Existential unforgeability for N parties). *An N-party adaptor signature scheme* $\mathsf{N\text{-}AS}_{R,\Sigma}$ *is* $\mathsf{N\text{-}aEUF\text{-}CMA}$ *secure if for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \dots, \mathcal{A}_{N-1})$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\sum_{i=1}^{N-2} \Pr[\mathsf{N\text{-}aSigForge}_{\mathcal{A}_i,\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda) = 1] + \Pr[\mathsf{N\text{-}aSigForge}_{\mathcal{A}_{N-1},\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda) = 1] \\ \leq \mathsf{negl}(\lambda)$$

where the experiments $\mathsf{N\text{-}aSigForge}_{\mathcal{A}_i,\mathsf{N\text{-}AS}_{R,\Sigma}}$ *and* $\mathsf{N\text{-}aSigForge}_{\mathcal{A}_{N-1},\mathsf{N\text{-}AS}_{R,\Sigma}}$ *are as defined in the Table 2.*

Theorem 2. *If the Schnorr signature scheme* Σ_{Sch} *is* $\mathsf{SUF\text{-}CMA}$ *secure, and* R_g *is a computationally hard algebraic relation, then* $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$ *in Fig. 5 is secure in the random oracle model.*

Regarding existential unforgeability for N parties, the following lemma holds.

Lemma 2 ($\mathsf{N\text{-}aEUF\text{-}CMA}$ security.). *Assuming that Schnorr digital signature scheme* Σ_{Sch} *is* $\mathsf{SUF\text{-}CMA}$ *secure and* R_g *is a hard relation, the N-party adaptor signature scheme* $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}$*, as defined in Fig.5, is* $\mathsf{N\text{-}aEUF\text{-}CMA}$ *secure.*

We can apply similar case-by-case reasoning as in the three-party case. In the three-party scenario, we considered an attacker A_2 between U_2 and U_3 as case (i) and an attacker A_1 between U_1 and U_2 as case (ii). The same approach can be used for the N-party scenario: we consider an attacker A_{n-1} between U_{n-1} and U_n as case (i), and an attacker A_i between U_i and U_{i+1} as case (ii), with iteration over $1 \leq i \leq n-2$. This straightforward extension yields the desired results.

Table 2. Experiments $\mathsf{N\text{-}aSigForge}_{\mathcal{A}_i,\mathsf{N\text{-}AS}_{R,\Sigma}}$ and $\mathsf{N\text{-}aSigForge}_{\mathcal{A}_{N-1},\mathsf{N\text{-}AS}_{R,\Sigma}}$

```
$\mathsf{N\text{-}aSigForge}^{i}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$, $(\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1}) \leftarrow \mathsf{Gen}(1^\lambda)$
2 : $(Y_i, y_i) \leftarrow \mathsf{GenR}(1^\lambda)$
3 : $(M, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pA}(\cdot,\cdot,\cdot)}(\{\mathsf{pk}_j\}_{j=i+1}^{n}, \{Y_j\}_{j=i}^{n-1})$
4 : $\sigma_{i+1} \leftarrow \mathsf{PreAdapt}_{U_{i+1}}((Y_i, y_i), Y_{i-1}, \mathsf{pk}_{i+1}, (\mathsf{sk}_i, \mathsf{pk}_i), \sigma_{i+1}, M)$
5 : $\sigma_i \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pA}(\cdot,\cdot,\cdot)}(\{\sigma_j\}_{j=i+1}^{n}, \mathsf{st})$
6 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_{i+1}, \sigma_i, M))$
```

```
$\mathsf{N\text{-}aSigForge}^{n-1}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$, $(\mathsf{sk}_n, \mathsf{pk}_n) \leftarrow \mathsf{Gen}(1^\lambda)$
2 : $(Y_{n-1}, y_{n-1}) \leftarrow \mathsf{GenR}(1^\lambda)$
3 : $(M, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk}_n, Y_{n-1})$
4 : $\sigma_n \leftarrow \mathsf{PreSign}_{U_n}((\mathsf{pk}_n, \mathsf{sk}_n), Y_{n-1}, M)$
5 : $\sigma_{n-1} \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\sigma_n, \mathsf{st})$
6 : return $(M \notin Q \wedge \mathsf{Vrfy}_{U_n}(\mathsf{pk}_n, \sigma_{n-1}, M))$
```

```
$\mathcal{O}_{pA}(M, (Y_i, Y_{i+1}), \sigma_{i+2})$
1 : $\sigma_i \leftarrow \mathsf{PreAdapt}_{U_i}((Y_i, y_i), Y_{i-1}, \mathsf{pk}_{i+1}, (\mathsf{sk}_i, \mathsf{pk}_i), \sigma_{i+1}, M)$
2 : $Q := Q \cup \{M\}$
3 : return $\sigma_i$
```

```
$\mathcal{O}_S(M)$                                   $\mathcal{O}_{pS}(M, Y_{n-1})$
1 : $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_i, M)$     1 : $\sigma_n \leftarrow \mathsf{PreSign}(\mathsf{sk}_n, Y_{n-1}, M)$
2 : $Q := Q \cup \{M\}$                            2 : $Q := Q \cup \{M\}$
3 : return $\sigma_i$                               3 : return $\sigma_n$
```

5 Conclusion

In this paper, we explored a general extension of adaptor signature schemes that previously applied for two parties. First, we extended the two-party adaptor signature scheme to three parties and presented the security requirements that the extended scheme should satisfy. We also provided a specific construction example using Schnorr signatures. Then, we demonstrated that the resulting Schnorr-signature-based, three-party adaptor signature scheme satisfies all the defined security properties. Furthermore, by extending the scheme to N parties, we showed a general construction method for N-party adaptor signatures and again provided a specific construction example using Schnorr signatures. This illustrates the ease of extending from three to N parties.

References

1. Monero, `https://www.getmonero.org/`

2. Aumayr, L., Ersoy, O., Erwig, A., Faust, S., Hostáková, K., Maffei, M., Moreno-Sanchez, P., and Riahi, S.: *Generalized channels from limited blockchain scripts and adaptor signatures*, Asiacrypt (2021).
3. Aumayr, L., Thyagarajan, S. A., Malavolta, G., Moreno-Sanchez, P., and Maffei, M.: *Sleepy channels: Bi-directional payment channels without watchtowers*, ACM CCS (2022).
4. Bagaria, V., Neu, J., and Tse, D.: *Boomerang: Redundancy improves latency and throughput in payment-channel networks*, FC (2020).
5. Coalition for Content Provenance and Authenticity. [n. d.]. `https://c2pa.org/`.
6. C2PA Technical Specification, `https://c2pa.org/specifications/specifications/2.0/specs/C2PA_Specification.html`.
7. Chen, Y., Liu, J. N., Yang, A., Weng, J., Chen, M. R., Liu, Z., and Li, M.: *PAC-DAM: Privacy-Preserving and Adaptive Cross-Chain Digital Asset Marketplace*, IEEE Internet of Things Journal (2023).
8. Dai, W., Okamoto, T., and Yamamoto, G.: *Stronger security and generic constructions for adaptor signatures*, Indocrypt (2022).
9. Debris-Alazard, T., Sendrier, N., and Tillich, J.-P.: *Wave: A new code-based signature scheme*, Cryptology ePrint Archive, 2018/996 (2018).
10. Deshpande, A. and Herlihy, M.: *Privacy-preserving cross-chain atomic swaps*, FC (2020).
11. Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and Stehle, D.: *CRYSTALS–Dilithium: Digital Signatures from Module Lattices*, CHES (2018).
12. Erwig, A., Faust, S., Hostáková, K., Maitra, M., and Riahi, S.: *Two-party adaptor signatures from identification schemes*, PKC (2021).
13. Erwig, A. and Riahi, S.: *Deterministic Wallets for Adaptor Signatures*, ESORICS (2022).
14. Esgin, M. F., Ersoy, O., and Erkin, Z.: *Post-quantum adaptor signatures and payment channel networks*, ESORICS (2020).
15. Fiat, A. and Shamir, A.: *How to prove yourself: Practical solutions to identification and signature problems*, Eurocrypt (1986).
16. Gudgeon, L., Moreno-Sanchez, P., Roos, S., McCorry, P., and Gervais, A.: *Sok: Layer-two blockchain protocols*, FC (2020).
17. Han, R., Lin, H., and Yu, J.: *On the optionality and fairness of atomic swaps*, The 1st ACM Conference on Advances in Financial Technologies (2019).
18. Hoenisch, P. and del Pino, L. S.: *Atomic swaps between bitcoin and monero*, arXiv preprint arXiv:2101.12332 (2021).
19. Hoenisch, P., Mazumdar, S., Moreno-Sanchez, P., and Ruj, S.: *LightSwap: An Atomic Swap Does Not Require Timeouts at both Blockchains*, DPM (2022).
20. Hu, X., and Chen, H.: *Design and Analysis of an Anonymous and Fair Trading Scheme for Electronic Resources with Blind Adaptor Signature* 6th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI) (pp. 1-5). IEEE (2023).
21. Ji, Y., Xiao, Y., Gao, B., and Zhang, R.: *Threshold/Multi Adaptor Signature and Their Applications in Blockchains*, Electronics, 13(1), 76. 2023.
22. Klamti, J. B. and Hasan, M. A.: *Post-Quantum Two-Party Adaptor Signature Based on Coding Theory*, Cryptography (2022).
23. Liu, Z., Yang, A., Zeng, H., Jiang, C., and Ma, L.: *A Generalized Blockchain-Based Government Data Sharing Protocol*, Security and Communication Networks (2023).

24. Madathil, V., Thyagarajan, S. A., Vasilopoulos, D., Fournier, L., Malavolta, G., and Moreno-Sanchez, P.: *Cryptographic Oracle-Based Conditional Payments*, NDSS (2023).
25. Malavolta, G., Moreno-Sanchez, P., Schneidewind, C., Kate, A., and Maffei, M.: *Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability*, NDSS (2019).
26. Minaei, M., Chatzigiannis, P., Jin, S., Raghuraman, S., Kumaresan, R., Zamani, M., and Moreno-Sanchez, P.: *Unlinkability and Interoperability in Account-Based Universal Payment Channels*, FC (2023).
27. Mirzaei, A., Sakzad, A., Yu, J., and Steinfeld, R.: *Fppw: A fair and privacy preserving watchtower for bitcoin*, FC (2021).
28. Moreno-Sanchez, P., Blue, A., Le, D.V., Noether, S., Goodell, B., and Kate, A.: *DLSAG: non-interactive refund transactions for interoperable payment channels in monero*, FC (2020).
29. Poelstra, A.: *Scriptless Scripts. Presentation Slides*, `https://download.wpsoftware.net/bitcoin/wizardry/mw-slides/2017-05-milan-meetup/slides.pdf.` (accessed 2023-08-17).
30. Poelstra, A. and Nick, J.: *Adaptor Signatures and Atomic Swaps from Scriptless Scripts*, `https://github.com/ElementsProject/scriptless-scripts/blob/master/md/atomic-swap.md` (2017).
31. Qin, X., Pan, S., Mirzaei, A., Sui, Z., Ersoy, O., Sakzad, A., Esgin, M. F., Liu, J. K., Yu, J., and Yuen, T. H.: *Blindhub: Bitcoin-compatible privacy-preserving payment channel hubs supporting variable amounts*, IEEE S&P (2023).
32. Riahi, S., and Litos, O. S. T.: *Bitcoin Clique: Channel-free Off-chain Payments using Two-Shot Adaptor Signatures*, Cryptology ePrint Archive (2024).
33. Schnorr, C.-P.: *Efficient identification and signatures for smart cards*, CRYPTO (1990).
34. Shao, W., Wang, J., Wang, L., Jia, C., Xu, S., and Zhang, S.: *Auditable Blockchain Rewriting in Permissioned Setting with Mandatory Revocability for IoT*, IEEE Internet of Things Journal (2023).
35. Sui, Z., Liu, J. K., Yu, J., and Qin, X.: *Monet: A fast payment channel network for scriptless cryptocurrency monero*, IEEE DCS (2022).
36. Sui, Z., Liu, J. K., Yu, J., Au, M. H., and Liu, J.: *AuxChannel: Enabling efficient bi-directional channel for scriptless blockchains*, ACM AsiaCCS (2022).
37. Tairi, E., Moreno-Sanchez, P., and Maffei, M.: *A^2L: Anonymous atomic locks for scalability in payment channel hubs*, IEEE S&P (2021).
38. Tairi, E., Moreno-Sanchez, P. and Maffei, M.: *Post-quantum adaptor signature for privacy-preserving off-chain payments*, FC (2021).
39. Thyagarajan, S. A., Malavolta, G., Schmidt, F., and Schröder, D.: *Paymo: Payment channels for monero*, Cryptology ePrint Archive, 2020/1441 (2020).
40. Thyagarajan, S. A. K. and Malavolta, G.: *Lockable signatures for blockchains: Scriptless scripts for all signatures*, IEEE S&P (2021).
41. Tu, B., Zhang, M., and Yu, C.: *Efficient ECDSA-Based Adaptor Signature for Batched Atomic Swaps*, ISC (2022).
42. Wang, X., Lin, C., Huang, X., and He, D.: *Anonymity-Enhancing Multi-Hop Locks for Monero-Enabled Payment Channel Networks*, IEEE Transactions on Information Forensics and Security (2023).
43. Zhou, X., He, D., Ning, J., Luo, M., and Huang, X.: *Efficient Construction of Verifiable Timed Signatures and its Application in Scalable Payments*, IEEE Transactions on Information Forensics and Security (2023).

44. Zhu, X., He, D., Bao, Z., Peng, C., and Luo, M.: *Two-Party Adaptor Signature Scheme Based on IEEE P1363 Identity-Based Signature*, IEEE Open Journal of the Communications Society (2023).

A Comparison with Existing Multi-Party Settings

As explained above, our contribution lies in extending adaptor signatures to multi-party settings and investigating the gap with respect to two-party settings. Ji et al. [21] proposed a multi-adaptor signature scheme, but their multi-party setting differs from ours. In their setting, users who are performing pre-signatures exist "simultaneously" and the resulting n pre-signatures are aggregated into one via signature aggregation techniques before being sent to Alice, the secretary. Thus, Ji et al. considered n signers, and this extension could also be of interest in blockchain applications. In real-world applications, however, protocols that end in a single round trip are rare, and scenarios often involve routing or proxies, where someone else intervenes, or where Alice needs to forward messages to someone else. Therefore, our multi-party setting is more generalized. That is, given an initial pre-signer Bob, we consider the scenario of non-simultaneously receiving n users from Bob, who then passes on the pre-signatures sequentially. Eventually, the nth user performs adaptation to obtain a regular signature. At each handover, the execution of an "Extract" operation to obtain the secret witness enables the fair exchange desired in the two-party setting. Accordingly, we expect our approach to be applicable in data sharing and supply chain management, among other applications.

B Preliminaries

We first introduce the cryptographic primitives and notations used in this paper. We denote by $x \leftarrow X$ the uniform sampling of a variable x from a set X. Throughout this paper, λ denotes the security parameter, and all our algorithms run in polynomial time in λ. By writing $x \leftarrow A(y)$, we mean that, on input y, a probabilistic polynomial time (PPT) algorithm A outputs x. If A is a deterministic polynomial time (DPT) algorithm, then we use the notation $x := A(y)$. A function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ is negligible in n if, for every $k \subset N$, there exists $n_0 \in N$ s.t. for every $n \geq n_0$ it holds that $|\mathsf{negl}(n)| \leq 1/n^k$.

We next recall the definition of a hard relation R with statement/witness pairs (Y, y). Let L_R be the associated language defined as $\{Y | \exists y \text{ s.t.} (Y, y) \in R\}$. We say that R is a hard relation if the following hold: (i) There exists a PPT sampling algorithm GenR that, on input 1^λ, outputs a statement/witness pair $(Y, y) \in R$; (ii) the relation is poly-time decidable; and (iii) for all PPT A on input Y, the probability of A outputting a valid witness y is negligible.

B.1 Digital Signatures

A digital signature scheme Σ comprises the three algorithms KGen, Sign, and Vrfy. The key generation algorithm $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\lambda)$ takes a security parameter λ as an input and outputs a secret (signing) key sk and a public (verification) key pk. The signing algorithm $\sigma \leftarrow \mathsf{Sign}(m, \mathsf{pk}, \mathsf{sk})$ takes a message m, pk, and sk as inputs and outputs a signature σ. The verification algorithm $1/0 \leftarrow \mathsf{Vrfy}(m, \mathsf{pk}, \sigma)$ takes m, pk, and σ as inputs, and it outputs 1 if the signature is accepted, or 0 otherwise.

B.2 Schnorr signatures

In this section, we introduce one of the most fundamental signature schemes, Schnorr signatures, for later use in concrete configurations. Schnorr signatures are the most intuitive and most compatible signature scheme with adaptor signatures, because Poelstra [29] used them as a base when he first presented the concrete structure of adaptor signatures.

First, let $\mathbb{G} =< g >$ be a cyclic group of prime order q, and let $R_q \subset \mathbb{G} \times \mathbb{Z}_q$ be a relation defined as $R_q := \{(Y, y) | Y = g^y\}$, where $\mathbb{Z}_q$ is the set of integers modulo q.

Next, we briefly recall Schnorr signature scheme $\Sigma_{\mathrm{Sch}} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$. The key generation algorithm samples $x \leftarrow \mathbb{Z}_q$ uniformly at random and returns $X := g^x \in \mathbb{G}$ as the public key and x as the secret key. On an input message $m \in \{0,1\}^*$, the signing algorithm computes $r = \mathcal{H}(X||g^k||m) \in \mathbb{Z}_q$ and $s := k + rx \in \mathbb{Z}_q$, for $k \leftarrow \mathbb{Z}_q$ chosen uniformly at random, and it outputs a signature $\sigma := (r, s)$. Finally, on an input message $m \in \{0,1\}^*$ and signature $(r, s) \in \mathbb{Z}_q \times \mathbb{Z}_q$, the verification algorithm verifies that $r = \mathcal{H}(X||g^s \cdot X^{-r}||m)$.

B.3 Security Classes

Definition 3 (EUF-CMA). *A digital signature scheme Σ is considered an existentially unforgeable against adaptively chosen-message attack (EUF-CMA) secure if for any PPT adversary $\mathcal{A}$, $\Pr[\mathsf{SigForge}_{\mathsf{SIG},\mathcal{A}}(\lambda) = 1] = \mathsf{negl}(\lambda)$, where $\mathsf{SigForge}_{\mathsf{SIG},\mathcal{A}}(n)$ and $\mathcal{O}_S(\cdot)$ are defined in Fig. 2.*

```
$\mathsf{SigForge}_{\mathsf{SIG},\mathcal{A}}(\lambda)$:
    $\mathcal{Q} := \emptyset$
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^n)$
    $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_S(\cdot)}(\mathsf{pk})$
    If $m^* \in \mathcal{Q}$, then return 0
    Return $\mathsf{Vrfy}(\mathsf{pk}, m^*, \sigma^*)$
```

```
$\mathcal{O}_S(m)$:
    $\sigma \leftarrow \mathsf{Sign}(m, \mathsf{pk}, \mathsf{sk})$
    $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
    Return $\sigma$
```

Fig. 2. Experiment with EUF-CMA.

Definition 4 (SUF-CMA). *A digital signature scheme Σ is considered a strong existentially unforgeable against adaptively chosen-message attack (SUF-CMA) secure if for any PPT adversary $\mathcal{A}$, $\Pr[\mathsf{strongSigForge}_{\mathsf{SIG},\mathcal{A}}(\lambda) = 1] = \mathsf{negl}(\lambda)$, where $\mathsf{strongSigForge}_{\mathsf{SIG},\mathcal{A}}(\lambda)$ and $\mathcal{O}_S(\cdot)$ are defined in Fig. 3.*

```
$\mathsf{strongSigForge}_{\mathsf{SIG},\mathcal{A}}(\lambda)$:
$\mathcal{Q} := \emptyset$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_S(\cdot)}(\mathsf{pk})$
If $(m^*, \sigma^*) \in \mathcal{Q}$, then return 0
Return $\mathsf{Vrfy}(\mathsf{pk}, m^*, \sigma^*)$
```

```
$\mathcal{O}_S(m)$:
$\sigma \leftarrow \mathsf{Sign}(m, \mathsf{pk}, \mathsf{sk})$
$\mathcal{Q} := \mathcal{Q} \cup \{m, \sigma\}$
Return $\sigma$
```

Fig. 3. Experiment with SUF-CMA.

C Supplemental Material for Two-Party Adaptor Signatures

C.1 Syntax of Two-Party Adaptor Signatures

The adaptor signature $\mathsf{AS}_{R,\Sigma} = (\mathsf{PreSign}, \mathsf{PreVrfy}, \mathsf{Adapt}, \mathsf{Ext})$ is defined by four algorithms as follows. Note that the public key, private key, public information, and secret information used below are pre-prepared via $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(\lambda)$ and $(Y, y) \leftarrow \mathsf{GenR}(\lambda)$. First, the pre-signature generation algorithm $\hat{\sigma} \leftarrow \mathsf{PreSign}((\mathsf{pk}, \mathsf{sk}), Y, M)$ takes as input a pair of public and private keys $(\mathsf{pk}, \mathsf{sk})$, the public information Y, and a message M, and it outputs the pre-signature $\hat{\sigma}$. The pre-verification algorithm $1/0 \leftarrow \mathsf{PreVrfy}(Y, \mathsf{pk}, \hat{\sigma}, M)$ takes as input public information Y, the public key pk, the pre-signature $\hat{\sigma}$, and the message M, and it outputs 1 if the signature σ is accepted, or 0 otherwise. The adaptation algorithm $\sigma := \mathsf{Adapt}((Y, y), \mathsf{pk}, \hat{\sigma}, M)$ takes as input a pair comprising the public information Y and secret information y, the public key pk, the pre-signature $\hat{\sigma}$, and the message M, and it outputs a signature σ via a DPT algorithm. Finally, the extraction algorithm $y'/\perp \leftarrow \mathsf{Ext}(Y, \hat{\sigma}, \sigma)$ s.t. $(Y, y') \in R$ takes as input the public information Y, pre-signature $\hat{\sigma}$, and signature σ, and it outputs y' satisfying $(Y, y') \in R$ if $\hat{\sigma}$ and σ are correct, or $\perp$ otherwise. With adaptor signatures, a user who receives a pre-signature $\hat{\sigma}$ can obtain secret information from any $(\hat{\sigma}, \sigma)$ pair by adapting (Adapt) the secret information and pre-signature. For the security of the original two-party adaptor signatures, see the Appendix C.2. Dai et al [8] have proposed a revised security, but this paper considers it based on the original security.

C.2 Security of Two-Party Adaptor Signatures

Here, we introduce the security of adaptor signatures according to the definition by Aumayr et al. [2], which entails three properties. The adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ satisfies the following correctness:

Definition 5 (Pre-signature correctness). *For any message $M \in \{0,1\}^*$ and $(Y, y) \in R$, the adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ satisfies pre-signature cor-*

rectness if the following holds:

$$\Pr\left[\begin{array}{l|l} \mathsf{PreVrfy}(Y,\mathsf{pk},\hat{\sigma},M)=1; & (\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{Gen}(1^\lambda); \\ \mathsf{Vrfy}(\mathsf{pk},M,\sigma)=1; & \hat{\sigma}\leftarrow\mathsf{PreSign}((\mathsf{pk},\mathsf{sk}),Y,M); \\ (Y,y')\in R & \sigma:=\mathsf{Adapt}((Y,y),\mathsf{pk},\hat{\sigma},M); \\ & y':=\mathsf{Ext}(Y,\hat{\sigma},\sigma) \end{array}\right]=1.$$

Below, aEUF-CMA(existential unforgeability under chosen-message attack for adaptor signatures) is defined in terms of the EUF-CMA security of a general digital signature, by considering a scenario in which an additional pre-signature is provided for a randomly chosen public statement $Y \in L_R$. This first property of existential unforgeability aims to ensure the unforgeability of signatures even when the adversary has a pre-signature for a specific message M.

aEUF-CMA security protects the signer. It is similar to EUF-CMA for digital signatures but additionally requires that production of a forgery σ for some message M is hard even given a pre-signature on M with respect to a random statement $Y \in L_R$. Note that it is essential to allow the adversary to learn a pre-signature on M because, for practical applications, signature unforgeability needs to hold even when the adversary learns a pre-signature for M without knowing a witness for Y.

Definition 6 (Existential unforgeability). *For any probabilistic polynomial-time (PPT) algorithm* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, consider the experiment* $\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$ *in Table 3. If there exists a negligible function such that* $\Pr[\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, then the adaptor signature scheme* $\mathsf{AS}_{R,\Sigma}$ *is aEUF-CMA secure.*

Here, aEUF-CMA represents an for adaptively chosen-message attack under a chosen-message attack (CCA) security, and $\Pr[\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda) = 1]$ *represents the probability that the adversary* $\mathcal{A}$ *succeeds in the given experiment for the adaptor signature scheme* $\mathsf{AS}_{R,\Sigma}$ *with security parameter* λ*. The above inequality means that if this probability is non-negligible, then the scheme is not aEUF-CMA-secure.*

Table 3. Experiment of $\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$

```
$\mathsf{aSigForge}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$, $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2 : $(Y,y) \leftarrow \mathsf{GenR}(1^\lambda)$
3 : $(M,\mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk},Y)$
4 : $\hat{\sigma} \leftarrow \mathsf{PreSign}(\mathsf{sk},Y,M)$
5 : $\sigma \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\hat{\sigma},\mathsf{st})$
6 : return $(m \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk},\sigma,M))$
```

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

```
$\mathcal{O}_{pS}(M,Y)$
1 : $\hat{\sigma} \leftarrow \mathsf{PreSign}(\mathsf{sk},Y,M)$
2 : $Q := Q \cup \{M\}$
3 : return $\hat{\sigma}$
```

The third property, called pre-signature adaptability, protects the verifier. It guarantees that any valid pre-signature w.r.t. Y (possibly produced by a malicious signer) can be completed as a valid signature by using a witness y with $(Y, y) \in R$. Note that this property is stronger than the pre-signature correctness property, because we require that even pre-signatures that were not produced by $\mathsf{PreSign}$ but are valid can still be completed as into valid signatures.

Definition 7 (Pre-signature adaptability). *For any message $M \in \{0,1\}^*$, any statement/witness pair $(Y, y) \in R$, any public key pk and any pre-signature $\hat{\sigma} \in \{0,1\}^*$ with $\mathsf{PreVrfy}(\mathsf{pk}, M, Y; \hat{\sigma}) = 1$, an adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ satisfies pre-signature adaptability if we have $\mathsf{Vrfy}(M, \mathsf{pk}; \mathsf{Adapt}(\hat{\sigma}, y)) = 1$.*

The last property of interest is witness extractability, which protects the signer. Informally, it guarantees that a valid signature/pre-signature pair $(\sigma, \hat{\sigma})$ for a message/statement pair (m, Y) can be used to extract a witness y for Y. Hence, a malicious verifier cannot use a pre-signature $\hat{\sigma}$ to produce a valid signature σ without revealing a witness for Y.

Definition 8 (Witness extractability). *An adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ is witness extractable if, for every PPT adversary $A = (A_1, A_2)$, there exists a negligible function $\mathcal{V}$ such that the following holds: $Pr[aWitExt_{A,\mathsf{AS}_{R,\Sigma}}(n) = 1] \leq \mathcal{V}(\lambda)$, where the experiment $aWitExt_{A,\mathsf{AS}_{R,\Sigma}}$ is defined as in Table 4.*

Table 4. Experiment $\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$

```
$\mathsf{aWitExt}_{\mathcal{A},\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$
2 : $(M, Y, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot), \mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk})$
3 : $\hat{\sigma} \leftarrow \mathsf{PreSign}(\mathsf{sk}, Y, M)$
4 : $\sigma \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot), \mathcal{O}_{pS}(\cdot,\cdot)}(\hat{\sigma}, \mathsf{st})$
5 : return $(Y, \mathsf{Ext}(\mathsf{pk}, \sigma, \hat{\sigma}), T \notin R \wedge M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}, \sigma, M))$

$\mathcal{O}_S(M)$                                  $\mathcal{O}_{pS}(M, Y)$
1 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, M)$      1 : $\hat{\sigma} \leftarrow \mathsf{PreSign}(\mathsf{sk}, Y, M)$
2 : $Q := Q \cup \{M\}$                     2 : $Q := Q \cup \{M\}$
3 : return $\sigma$                          3 : return $\hat{\sigma}$
```

This security definition above does not explicitly consider the existential unforgeability of pre-signatures (*pre-signature existential unforgeability*). However, by considering an experiment that omits steps 4 and 5 in Definition 6 and directly returns the pre-signature $\hat{\sigma}$, we can effectively address this aspect. Given the similarity between this modified experiment and Definition 6, we omit the detailed description here.

Finally, given the above definitions, we have the following definition of an adaptor signature scheme's security.

Definition 9. *Suppose that a Schnorr signature scheme Σ_{Sch} is SUF-CMA and R_g is a hard relation. An adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ is considered secure* *when it satisfies pre-signature correctness, pre-signature adaptability, existential unforgeability, and witness extractability.*

D Supplemental Material for Three-Party Adaptor Signatures

D.1 Other Security Definitions

As in 2-party, a total of four security properties are required. Here we define the other three security properties, which are not mentioned in the main text. Regarding pre-signature adaptability for three-party adaptor signatures, as with unforgeability, we need to consider different cases depending on which signer is considered the malicious attacker. Here, we assume that only one entity, either U_2 or U_3, attempts to adapt the pre-signature. This is because, in three-party adaptor signatures with two consecutive dialogues, integrity between entities cannot be guaranteed if there are multiple attackers.

Finally, we consider an extension of witness extractability, as defined below. Here, we again perform a case analysis based on which of the three entities forges extraction of the secret information, specifically for U_1 or U_2 as the attacker. However, the entity performing such extraction is limited to one of U_1 or U_2. In this case, the attacker extracting the witness is an entity that receives the pre-signature, just as in the two-party case, so is U_3 never the attacker.

Definition 10 (Pre-signature correctness for three parties). *For any message $M \in \{0,1\}^*$ and $(Y_2, y_2), (Y_3, y_3) \in R$, the three-party adaptor signature scheme $\mathsf{AS}_{R,\Sigma}$ satisfies pre-signature correctness if the following holds:*

$$\Pr\left[\begin{array}{l|l} \mathsf{PreVrfy}_{U_2}(Y_2, \mathsf{pk}_3, \hat{\sigma}_3, M)=1; & (\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda); \\ \mathsf{Vrfy}(\mathsf{pk}_3, M, \sigma_2)=1; & (Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda); \\ (Y_2, y_2') \in R; & \hat{\sigma}_3 \leftarrow \mathsf{PreSign}_{U_3}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M); \\ \mathsf{PreVrfy}_{U_1}(Y_1, (\mathsf{pk}_2, \mathsf{pk}_3), & \hat{\sigma}_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \\ \qquad (\hat{\sigma}_2, \hat{\sigma}_3), M) = 1; & \qquad \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3, M); \\ \mathsf{Vrfy}(\mathsf{pk}_2, M, \sigma_1) = 1; & y_2' = \mathsf{PreExt}_{U_3}(Y_2, \hat{\sigma}_3, \hat{\sigma}_2); \\ (Y_1, y_1') \in R & \sigma_2 = \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \hat{\sigma}_2, M); \\ & y_1' / \perp = \mathsf{Ext}_{U_2}(Y_1, (\hat{\sigma}_2, \sigma_2)); \end{array}\right] = 1.$$

Definition 11 (Pre-signature adaptability for three parties). *For any message $M \in \{0,1\}^*$, any statement/witness pair $(Y_1, y_1), (Y_2, y_2) \in R$, and any key pairs $(\mathsf{pk}_2, \mathsf{sk}_2)$, $(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$, a three-party adaptor signature scheme 3-$\mathsf{AS}_{R,\Sigma}$ is pre-signature adaptable if following conditions (i) and (ii) below are satisfied.*
(i) If the sub-signer U_3 is an adversary, for any pre-signature $\hat{\sigma}_3 \leftarrow \{0,1\}^*$ with $\mathsf{PreVrfy}_{U_2}$ $(Y_2, \mathsf{pk}_3, \hat{\sigma}_3, M) = 1$, then we have

$$\mathsf{Vrfy}(\mathsf{pk}_3, M, \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3, M))=1.$$

(ii) If the main-signer U_2 is an adversary, for any pre-signatures $\hat{\sigma}_3 \leftarrow \mathsf{PreSign}_{U_3}($ $(\mathsf{pk}_3,\ \mathsf{sk}_3), Y_2, M)$ and $\hat{\sigma}_2 \leftarrow \{0,1\}^*$ with $\mathsf{PreVrfy}_{U_1}(Y_1, (\mathsf{pk}_2, \mathsf{pk}_3), (\hat{\sigma}_2, \hat{\sigma}_3), M) =$ 1, we have

$$\mathsf{Vrfy}(\mathsf{pk}_2, M, \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \hat{\sigma}_2, M)) = 1.$$

Definition 12 (Witness extractability for three parties). *A three-party adaptor signature scheme $\text{3-}\mathsf{AS}_{R,\Sigma}$ is witness extractable if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\Pr[\text{3-}\mathsf{aWitExt}_{\mathcal{A}_\infty,\text{3-}\mathsf{AS}_{R,\Sigma}}(\lambda) = 1] + \Pr[\text{3-}\mathsf{aWitExt}_{\mathcal{A}_\in,\text{3-}\mathsf{AS}_{R,\Sigma}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$$

where those experiments $\text{3-}\mathsf{aWitExt}_{\mathcal{A}_\infty,\text{3-}\mathsf{AS}_{R,\Sigma}}$ and $\text{3-}\mathsf{aWitExt}_{\mathcal{A}_\in,\text{3-}\mathsf{AS}_{R,\Sigma}}$ are as defined in Figure 4.

Fig. 4. Experiments for witness extractability

```
$\text{3-}\mathsf{aWitExt}_{\mathcal{A}_1,\text{3-}\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$
2 : $(\mathsf{sk}_2, \mathsf{pk}_2)(\mathsf{sk}_3, \mathsf{pk}_3) \leftarrow \mathsf{KeyGen}(1^\lambda)$
3 : $(M^*, Y_1, Y_2) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
4 : $\hat{\sigma}_3^* \leftarrow \mathsf{PreSign}_{U_3}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
5 : $\hat{\sigma}_2^* \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
6 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\hat{\sigma}_2^*, \hat{\sigma}_3^*, Y_1, Y_2)$
7 : $y_1' := \mathsf{Ext}(Y_1, \sigma_1, \hat{\sigma}_2^*)$
8 : $\mathbf{return}(M^* \notin Q \wedge (Y_1, y_1') \notin R \wedge \mathsf{Vrfy}(\mathsf{pk}_2, \sigma_1, M^*))$
```

```
$\text{3-}\mathsf{aWitExt}_{\mathcal{A}_2,\text{3-}\mathsf{AS}_{R,\Sigma}}(\lambda)$
1 : $Q := \emptyset$
2 : $(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
3 : $(M^*, Y_2) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot)}(\mathsf{pk})$
4 : $\hat{\sigma}_3^* \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
5 : $\hat{\sigma}_2 \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\hat{\sigma}_3^*)$
6 : $y_2' := \mathsf{Ext}(Y_2, \hat{\sigma}_2, \sigma_3^*)$
7 : $\mathbf{return}\ (M^* \notin Q \wedge (Y_2, y_2') \notin R' \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \hat{\sigma}_2, M^*))$
```

```
$\mathcal{O}_{pA}(Y_i, Y_{i+1}, M)$
1 : $\hat{\sigma}_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3, M)$
2 : $Q := Q \cup \{M\}$
3 : $\mathbf{return}\ \hat{\sigma}_2$
```

```
$\mathcal{O}_{pS}(M, Y_i)$
1 : $\hat{\sigma}_{i+1} \leftarrow \mathsf{PreSign}(\mathsf{sk}_{i+1}, Y_i, M)$
2 : $Q := Q \cup \{M\}$
3 : $\mathbf{return}\ \hat{\sigma}_{i+1}$
```

```
$\mathcal{O}_S^{\mathcal{A}_1}(M)$
1: $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_{i+1}, M)$
2: $Q := Q \cup \{M\}$
3: $\mathbf{return}\ \sigma_i$
```

```
$\mathcal{O}_S^{\mathcal{A}_2}(M)$
1: $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_{i+1}, M)$
2: $Q := Q \cup \{M\}$
3: $\mathbf{return}\ \sigma_i$
```

Given the above definitions, we have the following definition of three-party adaptor signature scheme's security.

Definition 13. *Suppose that Schnorr signature scheme Σ_{Sch} is SUF-CMA and R_g is a hard relation. $\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}$ is a secure three-party adaptor signature scheme in the ROM if three-party pre-signature correctness, three-party existential unforgeability, three-party pre-signature adaptability, and three-party witness extractability are satisfied.*

E Security Proof of Three-party Adaptor Signatures

Theorem 1. *If Schnorr signature scheme Σ_{Sch} is SUF-CMA-secure, and R_g is a computationally hard algebraic relation, then $\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$ in Fig. 1 is secure in the ROM.*

First, we prove the following useful proposition about reversibility of adaptor signatures.

Proposition 1 (Reversibility). *For any $\sigma_2 := (r_2, s_1) \in \mathbb{Z}_q \times \mathbb{Z}_q$ and any $y_1 \in \mathbb{Z}_q$,*

$$\mathsf{Adapt}_{U_1}(\mathsf{Adapt}_{U_1}((r_2, s_1), y_1), -y_1) = \sigma.$$

Proof of Proposition 1. By the definition of Adapt, for any $r_2, s_1, y_1 \in \mathbb{Z}_q$, we have

$$\begin{aligned}\mathsf{Adapt}_{U_1}(\mathsf{Adapt}_{U_1}((r_2, s_1), y_1), -y_1) &= \mathsf{Adapt}((r_2, s_1 + y_1), -y_1)\\ &= (r_2, s_1 + y_1 + (-y_1))\\ &= (r_2, s_1).\end{aligned}$$

In particular, this lemma implies that, by knowing a witness y, we can not only adapt a valid pre-signature with respect to g^{y_1} into a valid signature but also vice versa.

E.1 Proof of pre-signature correctness for three-party

Lemma 3 (Pre-signature correctness for three-party). *Schnorr-based adaptor signature scheme $\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}$ satisfies pre-signature correctness for three-party.*

Proof of Lemma 3. We show that the six equations of definition 5 are satisfied under the conditions of the definition.

– Correctness of $\mathsf{PreVrfy}_{U_2}(Y_2, \mathsf{pk}_3, \hat{\sigma}_3, M)$=1. Given $\hat{\sigma}_3 = (r_3, s_3)$ and $s_3 = k + r_3 x_3$,

$$g^{s_3} \cdot X_3^{-r_3} \cdot Y_2 = g^{k+r_3x_3} \cdot (g^{x_3})^{-r_3} Y_2 = g^k Y_2.$$

Therefore, $\mathsf{PreVrfy}_{U_2}(Y_2, \mathsf{pk}_3, \hat{\sigma}_3, M)$ computes

$$r_3 = \mathcal{H}(X_3||g^{s_3} \cdot g^k \cdot Y_2||M) = \mathcal{H}(X_3||g^{s_3} \cdot X_3^{-r_3} \cdot Y_2||M).$$

- Correctness of $\mathsf{PreVrfy}_{U_3}(Y_1, \mathsf{pk}_2, \hat{\sigma}_2, M) = 1$. Given $\hat{\sigma}_2 = (r_2, s_2, s'_3)$ and $s_2 = k' + r_2 x_2$,

$$g^{s_2} \cdot X_2^{-r_2} \cdot Y_1 = g^{k'+r_2x_2} \cdot (g^{x_2})^{-r_2} \cdot Y_1 = g^{k'} Y_1.$$

Therefore, $\mathsf{PreVrfy}_{U_3}(Y_1, \mathsf{pk}_2, \hat{\sigma}_2, M)$ computes

$$r_2 = \mathcal{H}(X_2 || g^{s_2} \cdot g^{k'} \cdot Y_1 || M) = \mathcal{H}(X_2 || g^{s_2} \cdot X_2^{-r_2} \cdot Y_1 || M).$$

- Correctness of $(Y_2, y'_2) \in R$. For $s'_3 = s_3 + y_2$, U_3 gets $y'_2 = s'_3 - s_3 = (s_3 + y_2) - s_3 = y_2$.

$$\therefore (Y_1, y'_1) \in R.$$

- Correctness of $\mathsf{PreVrfy}_{U_1}(Y_1, (\mathsf{pk}_2, \mathsf{pk}_3), (\hat{\sigma}_2, \hat{\sigma}_3), M)$=1. Given $\hat{\sigma}_2$=(r_2, s_2, s'_3) and $\hat{\sigma}_3 = (r_3, s_3)$, return 1 if $r_3 = \mathcal{H}(X_3 || g^{s_3} \cdot X_3^{-r_3} \cdot Y_2 || M)$ and $r_2 = \mathcal{H}(X_2 || g^{s_2} \cdot X_2^{-r_2} \cdot Y_1 || M)$ hold. The above can be demonstrated similarly to the correctness of $\mathsf{PreVrfy}_{U_2}$ and $\mathsf{PreVrfy}_{U_3}$.
- Correctness of $\mathsf{Vrfy}(\mathsf{pk}_2, M, \sigma_1)$=1. Given $\sigma_1 = (r_2, s_1)$,

$$\begin{aligned} g^{s_1} \cdot X_2^{-r_2} &= g^{s_2+y_1} \cdot (g^{x_2})^{-r_2} \ (\because s_1 = s_2 + y_1) \\ &= g^{k'+r_2x_2+y_1} \cdot (g^{x_2})^{-r_2} \ (\because s_2 = k' + r_2x_2) \\ &= g^{k'+y_1} \\ &= g^{k'} Y_1 \ (\because Y_1 = g^{y_1}). \end{aligned}$$

Therefore,

$$r_2 = \mathcal{H}(X_2 || g^{k'} Y_1 || M) = \mathcal{H}(X_2 || g^{s_1} X_2^{-r_2} || M).$$

- Correctness of $(Y_1, y'_1) \in R$. For $s_1 = s_2 + y_1$, U_2 gets $y'_1 = s_1 - s_2 = (s_2 + y_1) - s_2 = y_1$.

$$\therefore (Y_1, y'_1) \in R.$$

□

E.2 Proof of pre-signature adaptability for three-party

Lemma 2 (Pre-signature adaptability for three-party). *Schnorr-based adaptor signature scheme* $\mathsf{3\text{-}AS}_{R_g, \Sigma_{\mathrm{Sch}}}$ *satisfies pre-signature adaptability for three-party.*

Proof of Lemma 2. It is provable as well as the pre-signature adaptability of the 2-party. Two case divisions depending on which adaptability among the three parties is considered (i) when U_2 (main-signer) is adversary (ii) when U_3 (sub-signer) is adversary.

(i) If U_2 is an adversary, let $y_1, y_2 \in \mathbb{Z}_q$, $M \in \{0,1\}^*$, $\mathsf{pk}_2, \mathsf{pk}_3 \in \mathbb{G}$, and $\hat{\sigma}_2 = (r_2, \tilde{s}_2, s'_3) \in \mathbb{Z}_q \times \mathbb{Z}_q \times \mathbb{Z}_q$. Define $s_1 := \tilde{s}_2 + y_1$. Assume $\mathsf{PreVrfy}_{U_1}(Y_1, (\mathsf{pk}_2, \mathsf{pk}_3), (\hat{\sigma}_2, \hat{\sigma}_3), M) = 1$, then

$$\begin{aligned} r_2 &= \mathcal{H}(\mathsf{pk}_2 || g^{\tilde{s}_2} \cdot \mathsf{pk}_2^{-r_2} \cdot Y_1 || M) \\ &= \mathcal{H}(\mathsf{pk}_2 || g^{\tilde{s}_2} + y_1 \cdot \mathsf{pk}_2^{-r_2} || M) \\ &= \mathcal{H}(\mathsf{pk}_2 || g^{s_1} \cdot \mathsf{pk}_2^{-r_2} || M), \end{aligned}$$

which implies $\mathsf{Vrfy}(\mathsf{pk}_1, \sigma_1 = (r_2, s_1), M) = 1 (\because r_2 = \mathcal{H}(\mathsf{pk}_3 || g^{s_1} \cdot \mathsf{pk}_3^{-r_2} || M)$.

(ii) If U_3 is an adversary, let $y_1, y_2 \in \mathbb{Z}_q$, $M \in \{0,1\}^*$, $\mathsf{pk}_2, \mathsf{pk}_3 \in \mathbb{G}$, $\hat{\sigma}_3 = (r_3, \tilde{s}_3) \in \mathbb{Z}_1 \times \mathbb{Z}_q$. Define $s'_3 := \tilde{s}_3 + y_2$. Assume $\mathsf{PreVrfy}_{U_2}(Y_2, \mathsf{pk}_3, (r_3, \tilde{s}_3), M) = 1$, then

$$\begin{aligned} r_3 &= \mathcal{H}(\mathsf{pk}_3 || g^{\tilde{s}_3} \mathsf{pk}_3^{-r_3} Y_2 || M) \\ &= \mathcal{H}(\mathsf{pk}_3 || g^{\tilde{s}_3 + y_2} \mathsf{pk}_3^{-r_3} || M) \\ &= \mathcal{H}(\mathsf{pk}_3 || g^{s'_3} \mathsf{pk}_3^{-r_3} || M), \end{aligned}$$

which implies $\mathsf{Vrfy}(\mathsf{pk}_3, \hat{\sigma}_3 = (r_3, \hat{\sigma}_3), \hat{\sigma}_2 = (r_2, s_2, s'_3) || M) = 1$ since $r_3 = \mathcal{H}(\mathsf{pk}_3 || g^{s'_3} \mathsf{pk}_3^{-r_3} || M)$ holds.

E.3 Proof of unforgeability

Lemma 1 (3-aEUF-CMA security.). *Assuming that Schnorr digital signature scheme Σ_{Sch} is SUF-CMA secure and R_g is a hard relation, the three-party adaptor signature scheme $\mathsf{3\text{-}AS}_{R_g, \Sigma_{\mathrm{Sch}}}$, as defined in Fig.1, is 3-aEUF-CMA secure.*

A complete proof of Lemma lem: aEUF-CMA for 3 has already been given in section 3.3, but since the proof of Claim in each game hop is nontrivial, we prove each Claim here.

Supplemental Material for Proof of Lemma 3

Game $\mathbf{G_0}$: The game 0 is formally defined in Table 5 which corresponds to the original 3-aSigForge. From here, in the Table of security games, changes from the previous game are highlighted in yellow, and those that have already been changed are indicated in blue.

Game $\mathbf{G_1}$: Table 6 shows the difference game $\mathbf{G}_0$ and $\mathbf{G}_1$ of 3aEUF-CMA.

Claim of Game $\mathbf{G}_1$. Let Bad_1 be the event that $\mathbf{G}_1$ aborts. Then, $\Pr[\mathsf{Bad}_1] \leq v_1(\lambda)$, where v_1 is a negligible function in λ.

Proof of the Claim. Using adversary A_1, we construct a simulator S that solves a relation R_g and reduces it to a hard relation.

1. The simulator S generates a key pair $(\mathsf{sk}_2, \mathsf{pk}_2) \leftarrow \mathsf{Gen}(1^n)$ to simulate queries to the oracles $\mathcal{H}, \mathcal{O}_S, \mathcal{O}_{pS}, \mathcal{O}_{pA}$ of adversary A_1, where the oracles' behavior follows G_1.
2. Upon receiving a challenge message M^* from an adversary A, S calculates $\sigma_2 \leftarrow PreSign((pk_2, sk_2), Y_1^*, M^*)$ and returns a pair (σ_2, Y^*) to A_1.
3. When an adversary A outputs a forged signature σ_1^* and Bad_1 occurs (i.e., $Adapt(\sigma_2, y_1) = \sigma_1^*$), S can obtain $(Y_1^*, y_1^*) \in R$ by executing $y_1^* \leftarrow \mathsf{Ext}(\sigma_1^*, \sigma_2, Y_1^*)$ (because of pre-signature correctness).
4. A challenge public statement Y_1^* is an instance of the hard relation R and follows the output distribution of Gen_R. From the perspective of an adversary A, $Y_1 \approx Y_1^*$, and G_0 and G_1 are thus indistinguishable.

Therefore, the probability that the simulator S breaks the hard relation R_g is equal to the probability of Bad_1 occurring. □

Table 5. Formal definition of game $\mathbf{G_0}$

```
$\mathbf{G}_0$
1 : $Q := \emptyset$
2 : $\mathcal{H} := [\perp]$
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1^*, y_1^*)(Y_2^*, y_2^*) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot), \mathcal{O}_{pS}(\cdot), \mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2^*, M^*)$
7 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2^*, y_2^*),$
        $Y_1^*, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
8 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot), \mathcal{O}_{pS}(\cdot), \mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1^*, Y_2^*)$
9 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$

$\mathcal{O}_S^{\mathcal{A}_1}(M)$
1: $\sigma_1 \leftarrow \mathsf{Sign}((\mathsf{pk}_1, \mathsf{sk}_1), M)$
2: $Q := Q \cup \{M\}$
3: return $\sigma_1$

$\mathcal{O}_{pS}(M, Y_2)$
1: $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M)$
2: $Q := Q \cup \{M\}$
3: return $\sigma_3$

$\mathcal{O}_{pA}(M, (Y_2, Y_2), \sigma_3)$
1 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3,$
        $(\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M)$
2 : $Q := Q \cup \{M\}$
3 : return $\sigma_2$

$\mathcal{H}(x)$
1: if $\mathcal{H}[x] = \perp$
2: $\mathcal{H}[x] \leftarrow \mathbb{Z}_q$
3: return $\mathcal{H}[x]$
```

Game $\mathbf{G_2}$: Table 7 shows the difference game $\mathbf{G}_1$ and $\mathbf{G}_2$ of 3aEUF-CMA.

Claim of Game $\mathbf{G}_2$. Let Bad_2 be the event that $\mathbf{G}_2$ aborts in $\mathcal{O}_{pS}$. Then $\Pr[\mathsf{Bad}_2] \leq v_2(\lambda)$, where v_2 is a negligible function in n.

Proof of the Claim. $\mathsf{PreSign}$, Sign, and $\mathsf{PreAdapt}$ compute $K = g^k$ by using a uniformly random k from $\mathbb{Z}_q$. Bad_2 occurs when queries $\mathsf{pk}_3||K_3||M$ and $\mathsf{pk}_3||(K_3 \cdot Y_2)||M$ have not been generated before. As the adversary A is a PPT algorithm, the number of queries to each oracle $\mathcal{H}, \mathcal{O}_S, \mathcal{O}_{pS}, \mathcal{O}_{pA}$ is polynomial. The signature oracle $\mathcal{O}_S$, pre-signature oracle $\mathcal{O}_{pS}$, and pre-adaptation oracle $\mathcal{O}_{pA}$ use the above k when computing $K = g^k$. Let the numbers of queries to the oracles $\mathcal{H}, \mathcal{O}_S, \mathcal{O}_{pS}, \mathcal{O}_{pA}$ be l_1, l_2, l_3, l_4, respectively. Then,

$$\begin{aligned}\Pr[Bad_2] &= \Pr[\mathcal{H}'[pk_3||K_3||M] \neq \perp \vee \mathcal{H}'[pk_3||(K_3 \cdot Y_2)||M] \neq \perp] \\ &\leq \frac{2(l_1 + l_2 + l_3 + l_4)}{q} =: v_2'(\lambda),\end{aligned}$$

where l_1, l_2, l_3, l_4 are polynomials in λ, making v_2' is negligible. In total, there are $l_1 + l_2 + l_3 + l_4$ chances out of q, and two of them are for K and $K \cdot Y$. Therefore,

$$\Pr[G_1 = 1] \leq \Pr[G_2 = 1] + v_2'(\lambda).$$

□

Game $\mathbf{G_3}$: Table 8 shows the difference game $\mathbf{G}_2$ and $\mathbf{G}_3$ of 3aEUF-CMA.

Claim of Game $\mathbf{G}_3$. Let Bad_3 be the event that $\mathbf{G}_3$ aborts in $\mathcal{O}_{pA}$. Then $\Pr[\mathsf{Bad}_3] \leq v_3(\lambda)$, where v_3 is a negligible function in n.

Table 6. Difference between $\mathbf{G_0}$ and $\mathbf{G_1}$

```
G_1
1 : $Q := \emptyset$
2 : $\mathcal{H} := [\perp]$
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
7 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2),$
        $Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
8 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1, Y_2)$
9 : if $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) = \sigma_1^*$,
    abort.
10 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

Table 7. Difference between $\mathbf{G_1}$ and $\mathbf{G_2}$

```
$\mathcal{O}_{pS}(M, Y_2)$
1: $\mathcal{H}' := \mathcal{H}$
2: $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M)$
3: $(r_3, s_3) := \sigma_3$
4: $K := g^s \mathsf{pk}_3^{-r_3}$
5: if $\mathcal{H}'[\mathsf{pk}_3||K||M] \neq \perp$
6:     $\wedge\ \mathcal{H}'[\mathsf{pk}_3||K \cdot Y_2||M]$
7:   abort
8: $Q := Q \cup \{M\}$
9: return $\sigma_3$
```

Proof of the Claim. We first note that PreSign, Sign, and PreAdapt compute $K = g^k$ by choosing k uniformly at random from $\mathbb{Z}_q$. As A is PPT, the number of queries it can make to $\mathcal{H}$, $\mathcal{O}_{mathrmS}$, $\mathcal{O}_{\mathrm{pS}}$, and $\mathcal{O}_{\mathrm{pA}}$ is also polynomially bounded. Let l'_1, l'_2, l'_3, l'_4 be the numbers of queries made to $\mathcal{H}$, $\mathcal{O}_{mathrmS}$, $\mathcal{O}_{\mathrm{pS}}$, and $\mathcal{O}_{\mathrm{pA}}$ respectively. Furthermore, to address the case where A_1 uses $M(\neq M^*)$ and the challenge Y_2^*, S executes the algorithm $y'_1 \leftarrow \mathsf{Ext}(Y_1, \hat{\sigma}_2, \sigma_2)$. If the forged σ_2^* output by A_1 corresponds to a legitimate witness y_1^* derived from the challenge Y_1^* when $\sigma'_2 \leftarrow$ PreAdapt is computed using $M(\neq M^*)$ and the challenge Y_2^*, then the game aborts. As with G_1, this probability is bounded by the probability of breaking the hard relation, which is denoted as v_1. Then, we have the following:

$$\begin{aligned}\Pr[\mathsf{Bad}_3] &= \Pr[\mathcal{H}'[\mathsf{pk}_3||K_3||M] \neq \perp \wedge \mathcal{H}'[\mathsf{pk}_3||K_3 \cdot Y_2||M] \neq \perp] + v_1(\lambda)\\ &\leq 2\frac{l'_1 + l'_2 + l'_3 + l'_4}{q} + v_1(\lambda) = v'_3(\lambda) + v_1(\lambda),\end{aligned}$$

where v'_3 is a negligible function because l'_1, l'_2, l'_3, l'_4 are polynomial in λ and v_1 is the advantage of hard relation's advantage.

$$\therefore \Pr[G_2 = 1] \leq \Pr[G_3 = 1] + v_1(\lambda) + v'_3(\lambda). \tag{5}$$

□

Game $\mathbb{G}_4$: We note the difference between game $\mathbf{G_3}$ and game $\mathbf{G_4}$ in Table 9.

Game $\mathbb{G}_5$: We note the difference between game $\mathbf{G_4}$ and game $\mathbf{G_5}$ in Table 10.

Game $\mathbb{G}_6$: We note the difference between game $\mathbf{G_5}$ and game $\mathbf{G_6}$ in Table 11.

Table 8. Difference between game $\mathbf{G_2}$ and game $\mathbf{G_3}$

```
G_3
1 : $Q := \emptyset$, $S := \emptyset$
2 : $H := [\perp]$,
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
7 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2),$
          $Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
8 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1, Y_2)$
9 : if $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) = \sigma_1^*$,
10 : $\vee\ y_1^* \leftarrow \mathsf{Ext}(Y_1^*, \mathsf{PreAdapt}_{U_2}((Y_2^*, y_2^*), Y_1^*,$
          $\mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3^*, M), \sigma_2^*$
11 :    then abort
12 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

```
$\mathcal{O}_{pA}(M, (Y_2, Y_2), \sigma_3)$
1 : $\mathcal{H}' := \mathcal{H}$
2 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2), Y_1, \mathsf{pk}_3,$
          $(\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M)$
3 : $(r_2, s_2, s_3') := \sigma_2$
4 : $K_2 := g^{s_2} \cdot \mathsf{pk}_2^{-r_2}$
5 : $y_2' = s_3' - s_3$ $(\because (r_3, s_3) := \hat{\sigma}_3)$
6 : if $\mathcal{H}'[\mathsf{pk}_2||K_2||M] \neq \perp$
7 :     $\wedge\ \mathcal{H}'[\mathsf{pk}_2\ \ K_2 \cdot Y_1||M] \neq \perp$
8 :     $\wedge\ s_3' \in S$
9 :     $\wedge\ (Y_2^*, y_2') \in R^*$
10 :   abort
11 : $Q := Q \cup \{M\}$
12 : $S := S \cup \{s_3'\}$
13 : return $\sigma_2$
```

Table 9. Difference between game $\mathbf{G_3}$ and game $\mathbf{G_4}$

```
$\mathcal{O}_{pS}(M, Y_2)$
1: $\mathcal{H}' := \mathcal{H}$
2: $\sigma_3 \leftarrow \mathsf{Sign}((\mathsf{pk}_3, \mathsf{sk}_3), M)$
3: $(r_3, s_3) := \sigma_3$
4: $K := g^s \mathsf{pk}_3^{-r_3}$
5: if $\mathcal{H}'[\mathsf{pk}_3||K||M] \neq \perp$
6:     $\wedge \mathcal{H}'[\mathsf{pk}_3||K \cdot Y_2||M]$
7:   abort
8: $x := \mathsf{pk}_3||K_3||M$
9: $\mathcal{H}'[\mathsf{pk}_3||K_3 \cdot Y_2||M] := \mathcal{H}[x]$
10: $\mathcal{H}[x] \leftarrow \mathbb{Z}_q$
11: $Q := Q \cup \{M\}$
12: return $\sigma_3$
```

Table 10. Difference between game $\mathbf{G_4}$ and game $\mathbf{G_5}$

```
G_5
1 : $Q := \emptyset$, $S := \emptyset$
2 : $\mathcal{H} := [\bot]$,
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma_3 \leftarrow \mathsf{PreSign}((\mathsf{pk}_3, \mathsf{sk}_3), Y_2, M^*)$
7 : $\sigma_2 \leftarrow \mathsf{PreAdapt}_{U_2}((Y_2, y_2),$
        $Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
8 : $\sigma_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1, Y_2)$
9 : if $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) = \sigma_1^*$,
10 : $\vee\ y_1^* \leftarrow \mathsf{Ext}(Y_1^*, \mathsf{PreAdapt}_{U_2}((Y_2^*, y_2^*), Y_1^*,$
        $\mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3^*, M), \sigma_2^*$
11 :   then abort
12 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

```
$\mathcal{O}_{pA}(M, (Y_2, Y_2), \sigma_3)$
1 : $\mathcal{H}' := \mathcal{H}$
2 : $\sigma_2 \leftarrow \mathsf{Sign}((\mathsf{pk}_2, \mathsf{sk}_2), M)$
3 : $(r_2, s_2) := \sigma_2$
4 : $s_3' \leftarrow \mathbb{Z}_q$
5 : if $s_3' \in S$, abort
6 : $K_2 := g^{s_2} \cdot \mathsf{pk}_2^{-r_2}$
7 : $y_2' = s_3' - s_3$ $(\because (r_3, s_3) := \hat{\sigma}_3)$
8 : if $\mathcal{H}'[\mathsf{pk}_2 || K_2 || M] \neq \bot$
9 :     $\wedge\ \mathcal{H}'[\mathsf{pk}_2\ \ K_2 \cdot Y_1 || M] \neq \bot$
10 :     $\wedge\ s_3' \in S$
11 :     $\wedge\ (Y_2^*, y_2') \in R^*$
12 :   abort
13 : $x := \mathsf{pk}_2 || K_2 || M$
14 : $\mathcal{H}[\mathsf{pk2} || K_2 \cdot Y_1 || M] := \mathcal{H}[x]$
15 : $\mathcal{H}[x] \leftarrow \mathbb{Z}_q$
16 : $(r_2, s_2, s_3') := \hat{\sigma}_2$
17 : $Q := Q \cup \{M\}$
18 : $S := S \cup \{s_3'\}$
19 : return $\sigma_2$
```

Table 11. Difference between game $\mathbf{G_5}$ and game $\mathbf{G_6}$

```
G_6
1 : $Q := \emptyset$, $S := \emptyset$
2 : $\mathcal{H} := [\bot]$,
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma_2', \sigma_3' \leftarrow \mathsf{Sign}((\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3), M^*)$
7 : $(r_2', s_2') := \sigma_2', (r_3', s_3') := \sigma_3'$
8 : $\sigma_2 := \mathsf{Adapt}(\sigma_2', -y_1)$
9 : $\sigma_3 := \mathsf{Adapt}(\sigma_3', -y_2)$
        $Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
10 : $\sigma_1' \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma_2^*, \sigma_3^*, Y_1, Y_2)$
11 : if $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) = \sigma_1^*$,
12 : $\vee\ y_1^* \leftarrow \mathsf{Ext}(Y_1^*, \mathsf{PreAdapt}_{U_2}((Y_2^*, y_2^*), Y_1^*,$
        $\mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}_3^*, M), \sigma_2^*$
13 :   then abort
14 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

Table 12. Formal definition of game **strongSigForge**

```
$\mathcal{S}^{\mathsf{Sign}^{\mathsf{Sch}},\mathcal{H}^{\mathsf{Sch}}}$
1 : $Q := \emptyset$, $S := \emptyset$
2 : $\mathcal{H} := [\bot]$
3 : $(\mathsf{pk}_2, \mathsf{sk}_2)(\mathsf{pk}_3, \mathsf{sk}_3) \leftarrow \mathsf{Gen}(1^\lambda)$
4 : $(Y_1, y_1)(Y_2, y_2) \leftarrow \mathsf{GenR}(1^\lambda)$
5 : $M^* \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\mathsf{pk}_2, \mathsf{pk}_3)$
6 : $\sigma'_2, \sigma'_3 := \mathsf{Sign}^{\mathrm{Sch}}(M)$
7 : $(r'_2, s'_2) := \sigma'_2, (r'_3, s'_3) := \sigma'_3$
8 : $\sigma_2 := \mathsf{Adapt}(\sigma'_2, -y_1)$
9 : $\sigma_3 := \mathsf{Adapt}(\sigma'_3, -y_2)$
            $Y_1, \mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \sigma_3, M^*)$
10 : $\sigma'_1 \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot),\mathcal{O}_{pA}(\cdot)}(\sigma^*_2, \sigma^*_3, Y_1, Y_2)$
11 : if $\mathsf{Adapt}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) = \sigma^*_1$
12 : $\vee\ y^*_1 \leftarrow \mathsf{Ext}(Y^*_1, \mathsf{PreAdapt}_{U_2}((Y^*_2, y^*_2), Y^*_1,$
              $\mathsf{pk}_3, (\mathsf{sk}_2, \mathsf{pk}_2), \hat{\sigma}^*_3, M), \sigma^*_2$
13 :   then abort
14 : return $(M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_3, \sigma_2, M^*))$
```

```
$\mathcal{O}_{pA}(M, (Y_2, Y_2), \sigma_3)$
1 : $\mathcal{H}' := \mathcal{H}$
2 : $\sigma_2 := \mathsf{Sign}^{\mathrm{Sch}}(M)$
3 : $(r_2, s_2) := \sigma_2$
4 : $s'_3 \leftarrow \mathbb{Z}_q$
5 : if $s'_3 \in S$, abort
6 : $K_2 := g^{s_2} \cdot \mathsf{pk}_2^{-r_2}$
7 : $y'_2 = s'_3 - s_3$ $(\because (r_3, s_3) := \hat{\sigma}_3)$
8 : if $\mathcal{H}'[\mathsf{pk}_2 || K_2 || M] \neq \bot$
9 :     $\wedge\ \mathcal{H}'[\mathsf{pk}_2\ \ K_2 \cdot Y_1 || M] \neq \bot$
10 :    $\wedge\ s'_3 \in S$
11 :    $\wedge\ (Y^*_2, y'_2) \in R^*$
12 :  abort
13 : $x := \mathsf{pk}_2 || K_2 || M$
14 : $\mathcal{H}[\mathsf{pk}_2 || K_2 \cdot Y_1 || M] := \mathcal{H}^{\mathrm{Sch}}(x)$
15 : $\mathcal{H}[x] := \mathcal{H}^{\mathrm{Sch}}(\mathsf{pk}_2 || K_2 \cdot Y_1 || M)$
16 : $(r_2, s_2, s'_3) := \hat{\sigma}_2$
17 : $Q := Q \cup \{M\}$
18 : $S := S \cup \{s'_3\}$
19 : return $\sigma_2$
```

```
$\mathcal{O}_S^{\mathcal{A}_1}(M)$
1 : $\sigma_1 := \mathsf{Sign}^{\mathrm{Sch}}(M)$
2 : $(r_2, s_1) := \sigma_1$
3 : $K_1 := g^{s_1 \mathsf{pk}_2^{-r_2}}$
4 : $x := \mathsf{pk}_2 || K_1 || M$
5 : $\mathcal{H}[x] := \mathcal{H}^{\mathrm{Sch}}(x)$
6 : $Q := Q \cup \{M\}$
7 : return $\sigma_1$
```

```
$\mathcal{O}_{pS}(M, Y_2)$
1 : $\mathcal{H}' := \mathcal{H}$
2 : $\sigma_3 := \mathsf{Sign}^{\mathrm{Sch}}(M)$
3 : $(r_3, s_3) := \sigma_3$
4 : $K := g^s \mathsf{pk}_3^{-r_3}$
5 : if $\mathcal{H}'[\mathsf{pk}_3 || K || M] \neq \bot$
6 :     $\wedge \mathcal{H}'[\mathsf{pk}_3 || K \cdot Y_2 || M]$
7 :   abort
8 : $x := \mathsf{pk}_3 || K_3 || M$
9 : $\mathcal{H}[\mathsf{pk}_3 || K_3 \cdot Y_2 || M] := \mathcal{H}^{\mathrm{Sch}}(x)$
10 : $\mathcal{H}[x] := \mathcal{H}^{\mathrm{Sch}}(\mathsf{pk}_3 || X_3 \cdot Y_2 || M)$
11 : $Q := Q \cup \{M\}$
12 : return $\hat{\sigma}_3$
```

```
$\mathcal{H}(x)$
1: if $\mathcal{H}[x] = \bot$
2: $\mathcal{H}[x] \leftarrow \mathcal{H}^{\mathrm{Sch}}(x)$
3: return $\mathcal{H}[x]$
```

Game strongSigForge: We note the difference between game $\mathbf{G_6}$ and game **strongSigForge** in Table 12.

Claim of Game **strongSigForge***.* (M^*, σ^*) is a valid forgery of **strongSigForge**.

Proof of the Claim. We show that (M^*, σ^*) is never output by the oracle $\mathsf{Sign}^{\mathrm{Sch}}$. This proof follows similar reasoning to the two-party case. First, the simulated adversary has not made any queries to O_S, O_{pS}, or O_{pA} for the challenge message M^*. From Game G_1 and Proposition 1, the adversary outputs a forgery σ that is equal to a signature σ' output by $\mathsf{Sign}^{\mathrm{Sch}}$ during the challenge phase with only a negligible probability (i.e., the probability of breaking the hard relation). In this case, the simulation is aborted. Therefore, $\mathsf{Sign}^{\mathrm{Sch}}$ never outputs σ for M, and (M^*, σ^*) is thus a valid forgery for the game **strongSigForge**. □

E.4 Proof of witness extractability for three parties

Lemma 4 (Witness extractability for three parties.). *Assuming that* Σ_{Sch} *is* *SUF-CMA secure and* R_g *is a hard relation, the three-party adaptor signature scheme* $3\text{-}\mathsf{AS}_{R_g,\Sigma_{\mathrm{Sch}}}$*, as defined in Fig. 1, is witness extractable.*

As with the three-party unforgeability in Lemma 1, we reduce the witness extractability of the adaptor signature to the SUF-CMA security of Schnorr signature. That is, we demonstrate the existence of a simulator S when a PPT adversary $A = (A_1, A_2)$ wins the 3-aWitExt game, indicating successful forgery of the signature.

Because the PPT adversary acts as the witness extractor, in the case of two-party witness extractability, we consider the scenario where a user corresponding to the secretary becomes the adversary. Under the three-party conditions, however, either the secretary U_1, who may receive the public statement, or the main signer U_2, can potentially become the adversary. Therefore, as with Lemma 1, we consider two cases (i) and (ii).

The strategy involves sending the adversary a complete signature and behaving as if that signature were a valid pre-signature. However, in the pre-signature simulation in aWitExt, unlike in aSigForge, the adversary outputs the public statement Y along with the message M, so the game does not select pairs (Y, y). Therefore, S cannot convert a full signature to a pre-signature by executing $\mathrm{Adapt}(\sigma, -y)$ with the secret witness y.

Thus, to enable this transformation even without knowing y, we program a random oracle in which the values of $\mathcal{H}(g^x|K|m)$ and $\mathcal{H}(g^x|KY|m)$ are swapped (where the simulator knows $K = g^k$, g^x, and Y). Here, if at least one of $g^x|K|m$ or $g^x|KY|m$ has been queried to H before, the random oracle cannot be programmed. However, because A is PPT and k is uniformly randomly chosen from $\mathbb{Z}_q$, the probability that these values have been queried to H before is negligibly low.

Proof of Lemma 4. As noted above, we divide Lemma 4 into two cases, (i) and (ii), and we perform several game hops in each case to prove the lemma. For case (i), we can follow a similar procedure to the proof of unforgeability in the original two-party scenario in Lemma 5 of [2] and in Lemma 1. Hence, we demonstrate each game hop and reduction efficiency as follows.

Case (i). For case (i), we have the following game definitions for $\mathbf{G}_0$ to $\mathbf{G}_4$ and $\mathsf{strongSigForge}$.

Game $\mathbf{G}_0$: A three-party adaptor signature game, $\mathsf{3\text{-}aWitExt}_{\mathcal{A}_2,\mathsf{AS}_{R_g,\Sigma}}$.
Game $\mathbf{G}_1$: An abort game for when queries to the oracle from A_2 overlap for $H'[\mathsf{pk}||K||M]$ and $H'[\mathsf{pk}||K \cdot Y||M]$
Game $\mathbf{G}_2$: A game where the pre-signature oracle returns a regular signature
Game $\mathbf{G}_3$: A game where the same modifications as for $\mathbf{G_1}$ are applied to S. The game aborts if S has already queried $H^{'}[\mathsf{pk}||K||M]$ and $H^{'}[\mathsf{pk}||K{\cdot}Y||M]$.
Game $\mathbf{G}_4$: A game where the same modifications as for $\mathbf{G_2}$ are applied to S. The game passes regular signatures to A instead of pre-signatures.
Game $\mathsf{strongSigForge}$: A $\mathsf{SUF\text{-}CMA}$ game for regular signatures.

The reduction loss for the above is as follows and can be directly obtained from the original two-party scenario:

$$\begin{aligned}
&\Pr[\mathsf{3\text{-}aWitExt}_{\mathcal{A}_2,\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}}(\lambda) = 1] \\
&= \Pr[\mathbf{G}_0 = 1] \\
&\leq \Pr[\mathbf{G}_1 = 1] + v(\lambda) \\
&= \Pr[\mathbf{G}_2 = 1] + v(\lambda) \\
&= \Pr[\mathbf{G}_3 = 1] + v(\lambda) + v(\lambda) \\
&\leq \Pr[\mathbf{G}_4 = 1] + 2v(\lambda) \\
&= \Pr[\mathsf{strongSigForge}_{S^{A_2},\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}}(\lambda) = 1] + 2v(\lambda),
\end{aligned} \tag{6}$$

where v is a negligible function in λ.

Case (ii). For case (ii), we have the following game definitions for $\mathbf{G}_0$ to $\mathbf{G}_6$ and $\mathsf{strongSigForge}$.

Game $\mathbf{G}_0$: A three-party adaptor signature game, $\mathsf{3\text{-}aWitExt}_{\mathcal{A}_1,\mathsf{AS}_{R_g,\Sigma}}$. The changes from $\mathsf{3\text{-}aWitExt}$ to G_0 involve recording the hash values in the game's random oracle, namely the addition of $H := [\perp]$. The reduction between the games is $\Pr[\mathsf{3\text{-}aWitExt}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}}(\lambda) = 1] = \Pr[\mathbf{G}_0 = 1]$.
Game $\mathbf{G}_1$: A game that aborts if queries to the pre-signature oracle from A_2 overlap for $H'[\mathsf{pk}_2||K||M]$ and $H'[\mathsf{pk}_2||K \cdot Y_1||M]$. The modification from G_0 to G_1 is that, in the pre-signature oracle $\mathcal{O}_{pS}$, if either the pre-signature's format or the normal Schnorr signature's format has already been queried to the random oracle $\mathcal{H}$, the game aborts. Let Bad_1 denote the event that G_1 aborts. Let ℓ denote the total number of queries to each oracle. Because

ℓ is a polynomial in λ and v is a negligible function, the reduction between the games is as follows:

$$\begin{aligned}\Pr[\mathsf{Bad}_1] &= \Pr[H'[\mathsf{pk}_2||K||M] \neq\perp \wedge H'[\mathsf{pk}_2||K \cdot Y_1||M] \neq\perp] \\ &\leq 2\frac{\ell}{q} := v_1(\lambda),\end{aligned}$$

where v_1 is negligible in λ. Therefore, we obtain $\Pr[G_0 = 1] \leq \Pr[G_1 = 1] + v_1(\lambda)$.

Game $\mathbf{G}_2$: A game that aborts if queries from A to the pre-adaptation oracle overlap. The modification from G_1 to G_2 is that, in the pre-adaptation oracle $\mathcal{O}_{pA}$, if either the pre-signature's format or the normal Schnorr signature's format has already been queried to the random oracle $\mathcal{H}$, the game aborts. Similarly to G_1, the reduction between the games is as follows:

$$\Pr[G_1 = 1] \leq \Pr[G_2 = 1] + v_2(\lambda),$$

where v_2 is negligible in λ.

Game $\mathbf{G}_3$: A game whose pre-signature oracle returns a regular signature. In this game G_3, the pre-signature oracle $\mathcal{O}_{pS}$ outputs the signature σ instead of a pre-signature $\tilde{\sigma}$. The value of $H[\mathsf{pk}||K||M]$ is set to $H[\mathsf{pk}||K \cdot Y||M]$, and $H[\mathsf{pk}||K||M]$ is set to a new random value by the random oracle.
The adversary A cannot distinguish full signatures and pre-signatures, but it can only notice the change in this game if the random oracle has been queried on either $\mathsf{pk}||K||M$ or $\mathsf{pk}||K \cdot Y||M$. However, as this case is covered in G_1, the reduction loss is $\Pr[G_2 = 1] = \Pr[G_3 = 1]$.

Game $\mathbf{G}_4$: A game where the pre-adaptation oracle returns regular signatures. In this game G_4, the pre-adaptation oracle $\mathcal{O}_{pA}$ generates a regular signature σ instead of a pre-signature $\tilde{\sigma}$. However, because this oracle's form differs from that of a pre-signature oracle, some adjustments are necessary. These adjustments can be made similarly to those for the unforgeability game G_3. First, the value of $H[\mathsf{pk}||K||M]$ is set to $H[\mathsf{pk}||K \cdot Y||M]$, and $H[\mathsf{pk}||K||M]$ is set to a new random value by the random oracle. As in Game G_3 above, the adversary A cannot distinguish full signatures and pre-signatures but can only notice the change in this game if the random oracle has been queried on either $\mathsf{pk}||K||M$ or $\mathsf{pk}||K \cdot Y||M$. Again, this case is covered in G_1.
Next, the adversary's output forged witness y'_2 is checked against the challenge statement Y^*_2 (by chance) or on the oracle side. If y'_2 does not correspond to Y^*_2, the game aborts. The probability here is equal to the probability of breaking the hard relation v_3. Therefore, the reduction loss is $\Pr[G_3 = 1] \leq \Pr[G_4 = 1] + v_3(\lambda)$, where v_3 is negligible in λ.

Game $\mathbf{G}_5$: A game that applies the same modifications as in $\mathbf{G_1}$ and $\mathbf{G}_2$ to S. The game aborts if S has already queried $H^{'}[\mathsf{pk}||K||M]$ and $H^{'}[\mathsf{pk}||K{\cdot}Y||M]$. The reduction loss is $\Pr[G_4 = 1] \leq \Pr[G_5 = 1] + v_1(\lambda) + v_2(\lambda)$.

Game $\mathbf{G}_6$: A game that applies the same modifications as in $\mathbf{G}_3$ and $\mathbf{G}_4$ to S. The game passes regular signatures instead of pre-signatures to A. The reduction loss is $\Pr[G_5 = 1] \leq \Pr[G_6 = 1] + v_3(\lambda)$.

Game $\mathsf{strongSigForge}$**:** The $\mathsf{SUF\text{-}CMA}$ game for regular signatures. In this game, unlike in $\mathsf{3\text{-}aEUF\text{-}CMA}$ security, the adversary A_1 outputs a message M^* and public statements Y_1^*, Y_2^* as the challenge messages. Therefore, to convert full signatures into pre-signatures, the random oracle's output is programmed such that full signatures become regular signatures and vice versa. As a result, although regular Schnorr signatures are sent to the adversary, it can behave as if those signatures are valid pre-signatures, thus fully simulating Game G_6. Note that we will give a formal, detailed proof of this in the full version of this paper. Hence, the reduction is $\Pr[G_6 = 1] = \Pr[\mathsf{strongSigForge} = 1]$.

Finally, from the above discussion, the reduction efficiency for case (ii) is as follows:

$$\begin{aligned}
&\Pr[\mathsf{3\text{-}aWitExt}_{\mathcal{A}_1,\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}}(\lambda) = 1] \\
&= \Pr[\mathbf{G}_0 = 1] \\
&\le \Pr[\mathbf{G}_1 = 1] + v_1(\lambda) \\
&\le \Pr[\mathbf{G}_2 = 1] + v_1(\lambda) + v_2(\lambda) \\
&= \Pr[\mathbf{G}_3 = 1] + v_1(\lambda) + v_2(\lambda) \\
&\le \Pr[\mathbf{G}_4 = 1] + v_1(\lambda) + v_2(\lambda) + v_3(\lambda) \\
&\le \Pr[\mathbf{G}_5 = 1] + 2v_1(\lambda) + 2v_2(\lambda) + v_3(\lambda) \\
&\le \Pr[\mathbf{G}_6 = 1] + 2v_1(\lambda) + 2v_2(\lambda) + 2v_3(\lambda) \\
&= \Pr[\mathsf{strongSigForge}_{S^{A_1},\mathsf{3\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}}(\lambda) = 1] + 2v_1(\lambda) + 2v_2(\lambda) + 2v_3(\lambda).
\end{aligned}$$

$\square$

F Supplemental Material for N-Party Adaptor Signatures

F.1 Concrete Construction of Schnorr-Based N-Party Adaptor Signatures

Here, we extend the three-party adaptor signature scheme defined in Section 3 to describe a specific instantiation of the Schnorr-based N-party adaptor signature scheme given in Fig. 5. For Schnorr signatures Σ_{Sch} and a hard relation $R_g := \{(Y, y) | Y = g^y\}$, we show the concrete construction of the N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$.

F.2 Formal Security Definitions

The N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R,\Sigma}$ satisfies the following correctness.

Definition 14 (Pre-signature correctness for N parties). *For any message $M \in \{0,1\}^*$ and $(Y_1, y_1) \dots (Y_{n-1}, y_{n-1}) \in R$, the N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R,\Sigma}$ satisfies pre-signature correctness for N parties if the following holds:*

Fig. 5. Concrete construction: Schnorr-based N-party adaptor signatures.

U_1: $\underline{(Y_1, y_1) \leftarrow \mathsf{GenR}(\lambda);}$

$y_1 \leftarrow \mathbb{Z}_q^*$, $Y_1 := g^{y_1}$,
return Y_1 to U_2.

U_2: $\underline{(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KeyGen}(\lambda),}$
$\underline{(Y_i, y_i) \leftarrow \mathsf{GenR}(\lambda);}$

$\mathsf{sk}_i := x_i \leftarrow \mathbb{Z}_q$, $\mathsf{pk}_i = X_2 := g^{x_i} \in \mathbf{G}$,
$y_i \leftarrow \mathbb{Z}_g^*$, $Y_i := g^{y_i}$,
return Y_i to U_{i+1} and pk_i to U_{i-1}, U_{i+1}.

U_n: $\underline{(\mathsf{sk}_n, \mathsf{pk}_n) \leftarrow \mathsf{KGen}(\lambda);}$

$\mathsf{sk}_n := x_n \leftarrow \mathbb{Z}_q^*$, $\mathsf{pk}_n = X_n := g^{x_n} \in \mathbf{G}$,
return pk_n to U_i.

U_n: $\underline{\hat{\sigma}_n \leftarrow \mathsf{PreSign}_{U_n}((\mathsf{pk}_n, \mathsf{sk}_n), Y_{n-1}, M);}$

$k_n \leftarrow \mathbb{Z}_q$, $r_n := \mathcal{H}(X_n || g^{k_n} Y_{n-1} || M)$,
$s_n := k_n + r_n \cdot x_n$, $\hat{\sigma}_n := (r_n, s_n)$,
return $(\hat{\sigma}_n, M)$ to U_{n-1}.

U_i: $(1 \leq i \leq n-1)$:
$\underline{0/1 \leftarrow \mathsf{PreVrfy}_{U_i}\left(\{Y_j\}_{j=i}^{n-1}, \{\mathsf{pk}_j\}_{j=i+1}^{n}, \{\sigma_j\}_{j=i+1}^{n}, M\right);}$

For $1 \leq i \leq n-1$, $i+1 \leq j \leq n$,

return 1 if $r_j = \mathcal{H}\left(X_j || g^{s_j} \cdot X_j^{-r_j} \cdot Y_{j-1} || M\right)$.

$U_i(2 \leq i \leq n-1)$:
$\underline{\hat{\sigma}_i \leftarrow \mathsf{PreAdapt}_{U_i}\left((Y_i, y_i), Y_{i-1}, \mathsf{pk}_{i+1}, (\mathsf{sk}_i, \mathsf{pk}_i), \sigma_{i+1}, M\right);}$

$k_i \leftarrow \mathbb{Z}_q$, $r_i := \mathcal{H}\left(X_i || g^{k_i} Y_{i-1} || M\right)$,
$s_i := k_i + r_i \cdot x_i$, $s'_{i+1} := s_{i+1} + y_i$,
$\hat{\sigma}_i = (r_i, s_i, s'_{i+1})$,
return $\{\hat{\sigma}_j\}_{j=i}^{n}$ and M to U_{i-1} and $\hat{\sigma}_i$ to U_{i+1}.

U_1: $\underline{\sigma_1 = \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \hat{\sigma}_2, M);}$

$s_1 := s_2 + y_1$, $\sigma_1 := (r_2, s_1)$,
return σ_1 to U_2.

U_2: $\underline{y'_1 / \perp = \mathsf{Ext}_{U_2}(Y_1, (\hat{\sigma}_2, \sigma_1));}$

$y'_1 := s_1 - s_2$,
return y'_1, If $(Y_1, y'_1) \in R$, otherwise,
return $\perp$.

$U_i(2 < i \leq n)$:
$\underline{y'_{i-1} / \perp = \mathsf{PreExt}_{U_i}(Y_{i-1}, \hat{\sigma}_i, \hat{\sigma}_{i-1});}$

For $2 < i \leq n$, $y'_{i-1} := s'_i - s_i$.
return y'_{i-1}, if $(Y_{i-1}, y'_{i-1}) \in R$,
otherwise, return $\perp$.

$$\Pr\left[\begin{array}{l|l} & \{\mathsf{sk}_j, \mathsf{pk}_j\}_{j=2}^{n} \leftarrow \mathsf{Gen}(1^\lambda); \\ \mathsf{PreVrfy}_{U_i}(\{Y_j\}_{j=i}^{n-1}, & \{Y_j, y_j\}_{j=1}^{n-1} \leftarrow \mathsf{GenR}(1^\lambda); \\ \quad \{\mathsf{pk}_j, \sigma_j\}_{j=i+1}^{n}, M) = 1; & \sigma_n \leftarrow \mathsf{PreSign}_{U_n}((\mathsf{pk}_n, \mathsf{sk}_n), Y_{n-1}, M); \\ \mathsf{Vrfy}(\mathsf{pk}_i, M, \sigma_{i-1}) = 1; & \sigma_i \leftarrow \mathsf{PreAdapt}_{U_i}((Y_i, y_i), Y_{i-1}, \\ (Y_{i-1}, y'_{i-1}) \in R & \qquad \mathsf{pk}_{i+1}, (\mathsf{sk}_i, \mathsf{pk}_i), \sigma_{i+1}, M); \\ & y_{i-1} := \mathsf{Ext}_{U_i}(Y_{i-1}, \sigma_i, \sigma_{i-1}); \\ & \sigma_1 := \mathsf{Adapt}_{U_1}((Y_1, y_1), \mathsf{pk}_2, \sigma_2, M) \end{array}\right] = 1.$$

Regarding pre-signature adaptability for N-party adaptor signatures, as with unforgeability, it is necessary to consider separate cases depending on which signer is considered as the malicious attacker. Here, we consider U_i $(2 \leq i \leq n)$ as the entity attempting to adapt the pre-signature. This is because N-party adaptor signatures entail $n-1$ consecutive interactions, so the legitimacy between entities cannot be guaranteed when two or more attackers are present.

Definition 15 (Pre-signature adaptability for N parties). *For any message $M \in \{0,1\}^*$, algebraic relation pairs $\{Y_j, y_j\}_{j=1}^{n-1} \in R$, and public keys $\{\mathsf{pk}_j\}_{j=2}^{n}$, the N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R,\Sigma}$ satisfies pre-signature adaptability for N parties if the following requirements hold. (i) When U_n is the attacker, for any randomly chosen pre-signature $\sigma_n \leftarrow \{0,1\}^*$ satisfying $\mathsf{PreVrfy}_{U_{n-1}}(\mathsf{pk}_n, M, Y_{n-1}, \sigma_n) = 1$, we have*

$$\mathsf{Vrfy}_{U_n}(M, \mathsf{pk}_n, \mathsf{PreAdapt}_{U_{n-1}}((Y_{n-1}, y_{n-1}), Y_{n-2}, \mathsf{pk}_n, (\mathsf{sk}_{n-1}, \mathsf{pk}_{n-1}), \sigma_n, M)) = 1.$$

(ii) When U_i $(2 \leq i < n)$ is the attacker, for any $\sigma_n \leftarrow \mathsf{PreSign}_{U_n}((\mathsf{pk}_n, \mathsf{sk}_n), Y_{n-1}, M)$ and $\sigma_i \leftarrow \{0,1\}^*$ satisfying $\mathsf{PreVrfy}_{U_i}(\{Y_j\}_{i}^{n-1}, \{\mathsf{pk}_j\}_{j=i+1}^{n}, \{\sigma_j\}_{j=i+1}^{n}, M) = 1$, the following conditions hold: for $2 < i < n$,

$$\mathsf{Vrfy}_{U_n}(M, \mathsf{pk}_n \mathsf{PreAdapt}_{U_{n-1}}((Y_{n-1}, y_{n-1}), Y_{n-1}, \mathsf{pk}_n, (\mathsf{sk}_{n-1}, \mathsf{pk}_{n-1}), \sigma_n, M)) = 1,$$

and for $i = 2$,

$$\mathsf{Vrfy}_{U_n}(M, \mathsf{pk}_i, \mathsf{Adapt}_{U_{i-1}}((Y_i, y_i), Y_{i-1}, \mathsf{pk}_i, \sigma_i, M)) = 1.$$

Next, we consider the extension of witness extractability. Again, depending on which entity among the N parties extracts secret information—i.e., when U_i is the attacker for $1 \leq i < n-1$, or when U_n is the attacker—we need to distinguish these cases. Note, however, that the entity extracting secret information is limited to the U_i $(1 \leq i \leq n-1)$.

Definition 16 (Witness extractability for N parties). *The N-party adaptor signature scheme $\mathsf{N\text{-}AS}_{R,\Sigma}$ is witness extractable if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\sum_{i=1}^{n-2} \Pr[\mathsf{N\text{-}aWitExt}_{\mathcal{A}_i, \mathsf{N\text{-}AS}_{R,\Sigma}}^{1 \leq i < n-1}(\lambda) = 1] + \Pr[\mathsf{N\text{-}aWitExt}_{\mathcal{A}_{N-1}, \mathsf{N\text{-}AS}_{R,\Sigma}}^{i=n-1}(\lambda) = 1]$$
$$\leq \mathsf{negl}(\lambda),$$

for experiments $\mathsf{N\text{-}aWitExt}^{1\leq i<n-1}_{\mathcal{A}_i,\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda)$ *and* $\mathsf{N\text{-}aWitExt}^{i=n-1}_{\mathcal{A}_{n-1},\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda)$.

Table 13. Experiment $\mathsf{N\text{-}aWitExt}_{\mathcal{A},\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda)$

```
$\mathsf{N\text{-}aWitExt}^{1\leq i<n-1}_{\mathcal{A}_i,\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda)$
1: $Q := \emptyset$, $(\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1}) \leftarrow \mathsf{Gen}(1^\lambda)$
2: $(M, Y_i, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pA}(\cdot,\cdot)}(\mathsf{pk}_{i+1}, Y_{i+1})$
3: $\sigma_{i+1} \leftarrow \mathsf{PreAdapt}_{U_{i+1}}((Y_{i+1}, y_{i+1}), Y_i, \mathsf{pk}_{i+2}, (\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1}), \sigma_{i+2}, M)$
4: $\sigma_i \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pA}(\cdot,\cdot)}(\sigma_{i+1}, \mathsf{st})$
5: $\textbf{return}(Y_i, \mathsf{Ext}_{U_{i+1}}(Y_i, \sigma_i, \sigma_{i+1}) \wedge M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_{i+1}, \sigma_i, M))$
```

```
$\mathsf{N\text{-}aWitExt}^{i=n-1}_{\mathcal{A}_{n-1},\mathsf{N\text{-}AS}_{R,\Sigma}}(\lambda)$
1: $Q := \emptyset$, $(\mathsf{sk}_{n-1}, \mathsf{pk}_{n-1}) \leftarrow \mathsf{Gen}(1^\lambda)$
2: $(M, Y_{n-1}, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk}_{n-1})$
3: $\hat{\sigma}_{n-1} \leftarrow \mathsf{PreSign}(\mathsf{sk}_{n-1}, Y_{n-1}, M)$
4: $\sigma_n \leftarrow \mathcal{A}_2^{\mathcal{O}_S(\cdot),\mathcal{O}_{pS}(\cdot,\cdot)}(\hat{\sigma}_{n-1}, \mathsf{st})$
5: $\textbf{return}\ (Y_{n-1}, \mathsf{Ext}_{U_n}(\sigma_n, \sigma_i, Y) \wedge M \notin Q \wedge \mathsf{Vrfy}(\mathsf{pk}_{n-1}, \sigma_n, M))$
```

```
$\mathcal{O}_{pA}(Y_i, Y_{i+1}, M)$
1: $\sigma_{i+1} \leftarrow \mathsf{PreAdapt}_{U_{i+1}}((Y_{i+1}, y_{i+1}), Y_i, \mathsf{pk}_{i+2}, (\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1}), \sigma_{i+2}, M)$
2: $Q := Q \cup \{M\}$
3: $\textbf{return}\ \sigma_{i+1}$
```

```
$\mathcal{O}_S(M)$
1: $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}_{i+1}, M)$
2: $Q := Q \cup \{M\}$
3: $\textbf{return}\ \sigma_i$
```

```
$\mathcal{O}_{pS}(M, Y_i)$
1 : $\sigma_{i+1} \leftarrow \mathsf{PreSign}(\mathsf{sk}_{i+1}, Y_i, M)$
2 : $Q := Q \cup \{M\}$
3 : $\textbf{return}\ \sigma_{i+1}$
```

Definition 17. *Suppose that Schnorr signature scheme Σ_{Sch} is SUF-CMA and R_g is a hard relation. $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathrm{Sch}}}$ is a secure three-party adaptor signature scheme in the ROM if N-party pre-signature correctness, three-party existential unforgeability, N-party pre-signature adaptability, and N-party witness extractability are satisfied.*

F.3 Security Proof

In this section, we demonstrate that N-party adaptor signatures based on Schnorr signatures satisfy the security properties defined above. Note that each proof is a straightforward extension of the corresponding proof given for the three-party case in Section 3.3.

Theorem 2. *If Schnorr signature scheme Σ_{Sch} is SUF-CMA secure, and R_g is a computationally hard algebraic relation, then $\mathsf{N\text{-}AS}_{R_g,\Sigma_{\mathsf{Sch}}}$ in Fig. 5 is secure in the random oracle model.*

To demonstrate the validity of Theorem 2, it suffices to show that it satisfies Definitions 14, 2, 15, and 16. Each of these properties has already been proven for the three-party case, and it is then trivial that they hold for the N-party case.

Regarding pre-signature correctness and pre-signature adaptability for N parties, it is sufficient to demonstrate that the two verification procedures Vrfy and $\mathsf{PreVrfy}_{U_i}$ hold in the N-party construction.

Finally, regarding existential unforgeability and witness extractability for N parties, we can apply similar case-by-case reasoning as in the three-party case. In the three-party scenario, we considered an attacker A_2 between U_2 and U_3 as case (i) and an attacker A_1 between U_1 and U_2 as case (ii). The same approach can be used for the N-party scenario: we consider an attacker A_{n-1} between U_{n-1} and U_n as case (i), and an attacker A_i between U_i and U_{i+1} as case (ii), with iteration over $1 \leq i \leq n-2$. This straightforward extension yields the desired results.