On the Security of Blind Signatures in the Multi-Signer Setting

Samuel Bedassa Alemu and Julia Kastner

Department of Computer Science
ETH Zurich, Switzerland
samuelbe@student.ethz.ch, julia.kastner@inf.ethz.ch

Abstract. Blind signatures were originally introduced by Chaum (CRYPTO '82) in the context of privacy-preserving electronic payment systems. Nowadays, the cryptographic primitive has also found applications in anonymous credentials and voting systems.
However, many practical blind signature schemes have only been analysed in the game-based setting where a single signer is present. This is somewhat unsatisfactory as blind signatures are intended to be deployed in a setting with many signers.
We address this in the following ways:

- We formalise two variants of one-more-unforgeability of blind signatures in the *Multi-Signer Setting*.
- We show that one-more-unforgeability in the *Single-Signer Setting* translates straightforwardly to the *Multi-Signer Setting* with a reduction loss proportional to the number of signers.
- We identify a class of blind signature schemes which we call *Key-Convertible* where this reduction loss can be traded for an increased number of signing sessions in the *Single-Signer Setting* and show that many practical blind signature schemes such as blind BLS, blind Schnorr, blind Okamoto-Schnorr as well as two pairing-free, ROS immune schemes by Tessaro and Zhu (Eurocrypt'22) fulfil this property.
- We further describe how the notion of *Key Substitution* attacks (Menezes and Smart, DCC'04) can be translated to blind signatures and provide a generic transformation of how they can be avoided.

1 Introduction

Blind Signature Schemes. In his seminal work titled "Blind Signature for Untraceable Payments" [10], David Chaum introduced the concept of blind signature and gave a general description regarding the defining characteristics of the scheme. He based his work on his concern about how payments were processed in the digital world. Chaum felt that electronic banking services had significant privacy implications. He argued that in traditional electronic payments, a third party keeps a record of the payee and the amount and the time of payment for each transaction, thus revealing a lot about the individual. Such payment systems contrast with anonymous payment systems such as banknotes and coins, which lack control and security and could potentially be used for illicit purposes. He proposed a new type of cryptography that allows payments to be made with more privacy whilst maintaining the same functionality of banknotes or electronic payments. In a later work [11], Chaum presented the first actual realisation of blind signatures, called Chaum's blind RSA scheme, a transformation of the classic RSA signature scheme [22].

Since then, (partially) blind signature schemes based on various assumptions have been presented, such as discrete logarithms [1, 2, 20, 24], pairings [7, 15], RSA [11], lattices [3, 17, 23]. Additionally, there are also generic constructions like [12] from other cryptographic building blocks.

Notably, apart from [12], most schemes have only been analysed in the game-based setting with a single signer.

The Multi-Instance Setting. Most game based cryptographic security definitions only take into account one game with a single challenger. As cryptographic primitives are deployed in the real world with many instances (e.g. many participants that each generate their own public key), considering only a single instance in security proofs may not be realistic. Hofheinz & Nguyen [18] define the *Multi-Instance Setting* generically for cryptographic primitives. Their interest is in the tightness of security proofs, i.e., the loss of advantage/increase in runtime that a reduction has compared to the adversary. They provide two variants of the *Multi-Instance Setting.* In both, the adversary is given access to n copies of the security game. In one setting, which we call the existential setting, the adversary then wins if it wins one of the copies, in the other, it wins if it wins all of the copies.

It is easy to see that a loose reduction can be given for the existential setting, as the reduction can merely guess which of the copies the adversary is going to win. The reduction loss is proportional to the number of copies of the game. We give the corresponding security definition for one-more unforgeability and the reduction in sections 3.2 and 4.2.

*Multi-Instance Setting*s have been investigated for various cryptographic primitives such as public key encryption [4], digital signature schemes [18, 19], lossy trapdoor functions [18].

Our Contribution. In this work, we extend the notion of One-More Unforgeability of blind signatures to the *Multi-Instance-Setting*, which we call the *Multi-Signer-Setting (MSS)*, as the adversary gets to interact with many signers. We give two definitions; one is *strong Multi-Signer OM-UF* where an adversary is required to provide a one-more forgery in one of the instances but has no requirement on the other instances (i.e., there may be instances where the adversary generates fewer signatures than completed signing sessions). For this security notion, we show that a straightforward reduction can be applied to show that security in the single-instance setting implies security in the *Multi-Instance Setting.* This reduction has a loss factor proportional to the number of signers, i.e., it is not tight.

The other setting we consider is *weak Multi-Signer OMUF* in which the adversary needs to output a forgery with respect to one of the instances but also is required to output *as many signatures as completed signing sessions* for all other instances. We identify a class of blind signature schemes that we call *key convertible* and show that security in the *Single-Instance Setting* tightly implies security in the weak *Multi-Instance Setting* for these schemes. The number of signing sessions needed in the single-instance setting is the sum of the number of signing sessions of all instances in the *Multi-Instance Setting.* We show that various existing blind signature schemes are *key convertible.*

Lastly, we consider a variant of the *Key Substitution* attack [19] for blind signatures. Informally speaking, in a *Key Substitution* attack, an adversary can come up with a new public key $\overline{\mathsf{pk}}$ for a message-signature pair (m, σ) that is valid with respect to pk, so that (m, σ) is also valid with respect to $\overline{\mathsf{pk}}$. Menezes & Smart [19] analysed this type of attack for (non-blind) digital signature schemes. This type of attack can be used in the digital signature setting to impersonate participants during an authenticated key exchange[6]. We give a definition of *Key Substitution* attacks for blind signatures and propose some transformations to prevent Key Substitution attacks.

2 Preliminaries

Notation. We denote by $s \xleftarrow{\$} S$ the uniform sampling of a value s from the finite set S, and by $y \xleftarrow{\$} \mathcal{A}(x_1, \ldots, x_n)$ that y is the output of a probabilistic algorithm $\mathcal{A}$ on inputs $x_1, \ldots, x_n$.

We denote by $\mathbb{Z}_p$ the ring of integers modulo $p \in \mathbb{Z}$, where usually p will be a prime.

We denote an interactive execution between algorithms X, and Y by $(a, b) \leftarrow \langle X(x), Y(y) \rangle$, where x is the private input of X and y is the private input of Y. The private output of X is a, and the private output of Y is b. We write $Y^{\langle X(x), \cdot \rangle^\infty}$ if Y can invoke an unbounded number of executions of the interactive protocol with X in arbitrary interleaved order. Accordingly, $X^{\langle \cdot, Y(y_0) \rangle^1, \langle \cdot, Y(y_1) \rangle^1}(x)$ can invoke arbitrarily ordered executions with $Y(y_0)$ and $Y(y_0)$ but interacts with each algorithm only once.

2.1 Groups

Definition 1 (Group Schemes). *A group scheme consists of the following algorithms:*

$\mathsf{Setup}()$ *: outputs group parameters* $\mathsf{pp}_{\mathbb{G}}$ *for a group* $\mathbb{G}$ *of prime order* p. *The group parameters contain a generator* g, *a neutral element, as well as the group order* p.

$\mathsf{Mult}(\mathsf{pp}_{\mathbb{G}}, g_1, g_2)$ *: takes as input two group elements and outputs* $g_1 \cdot g_2$ *(we will often write the latter to denote group multiplication)*

We require that the group axioms hold in $\mathbb{G}$ *with respect to* Mult.

Definition 2 (Discrete Logarithm Problem (DL)). *For a group scheme* $(\mathsf{Setup}, \mathsf{Mult})$*, we say that the discrete logarithm problem is* (t, ε)*-hard if for all probabilistic adversaries* $\mathcal{A}$ *that run in time at most* t *it holds that*

$$\mathsf{Adv}^{\mathrm{DLOG}}_{\mathcal{A}} = \Pr\left[x = x' : \begin{array}{c} \mathsf{pp}_{\mathbb{G}} \xleftarrow{\$} \mathsf{Setup}() \\ x \xleftarrow{\$} \mathbb{Z}_p \\ y := g^x \\ x' \xleftarrow{\$} \mathcal{A}(y) \end{array}\right] \leq \varepsilon$$

Definition 3 (Computational Diffie-Hellman Problem (CDH)). *For a group scheme* $(\mathsf{Setup}, \mathsf{Mult})$*, we say that the Computational Diffie-Hellman Problem is* (t, ε)*-hard if for all probabilistic adversaries* $\mathcal{A}$ *that run in time at most* t *it holds that*

$$\mathsf{Adv}^{\mathrm{CDH}}_{\mathcal{A}} = \Pr\left[z = g^{xy} : \begin{array}{c} \mathsf{pp}_{\mathbb{G}} \xleftarrow{\$} \mathsf{Setup}() \\ x, y \xleftarrow{\$} \mathbb{Z}_p \\ z \xleftarrow{\$} \mathcal{A}(g, g^x, g^y) \end{array}\right] \leq \varepsilon$$

Definition 4 (Decisional Diffie-Hellman Problem (DDH)). *For a group scheme* $(\mathsf{Setup}, \mathsf{Mult})$*, we say that the Decisional Diffie-Hellman Problem is* (t, ε)*-hard if for all probabilistic adversaries* $\mathcal{A}$ *that run in time at most* t *it holds that*

$$\mathsf{Adv}^{\mathrm{DDH}}_{\mathcal{A}} = 2\left|\Pr\left[b = b' : \begin{array}{c} \mathsf{pp}_{\mathbb{G}} \xleftarrow{\$} \mathsf{Setup}() \\ x, y, z \xleftarrow{\$} \mathbb{Z}_p \\ b' \xleftarrow{\$} \mathcal{A}(g, g^x, g^y, g^{xy+bz}) \end{array}\right] - \frac{1}{2}\right| \leq \varepsilon$$

Definition 5 (Gap Diffie-Hellman Groups). *A prime order group* $\mathbb{G}$ *is a Gap Diffie-Hellman (GDH) group if there exists an efficient algorithm* V_{DDH} *which solves the DDH problem in* $\mathbb{G}$ *and there is no polynomial-time algorithm which solves the CDH problem.*

2.2 Multi-Instance Settings

Definition 6 (Security Games [5]). *A security game consists of the following algorithms:* $\mathsf{Initialize}$, $\mathsf{Finalize}$ *and oracles* $O_1, \ldots O_m$*. For a game* Game *and an adversary* $\mathcal{A}$*, we denote by* $\mathsf{Game}^{\mathcal{A}}$ *the output of the game depicted in the left part of fig. 1.*

Remark 1. We will often denote the $\mathsf{Initialize}$ and $\mathsf{Finalize}$ algorithms as one algorithm Game which first runs $\mathsf{Initialize}$, then the adversary $\mathcal{A}$, and then $\mathsf{Finalize}$ to determine its output.

Definition 7 (Multi-Instance Settings [18]). *Let* Game *be a security game with a set of oracles* O*. We define the existential multi-instance setting of* Game *as described in the right side of fig. 1.*

```
$\mathsf{Game}^{\mathcal{A}}$
1 : $(C, \mathsf{state}) \xleftarrow{\$} \mathsf{Initialize}$
2 : $S \xleftarrow{\$} \mathcal{A}^{\mathsf{O}_1,\cdots,\mathsf{O}_m}(C)$
3 : return $\mathsf{Finalize}(C, \mathsf{state}, S)$
```

```
$\exists$-$n$-$\mathsf{Game}^{\mathcal{A}}$
1 : foreach $i \in [n]$
2 :     $(C_i, \mathsf{state}_i) \xleftarrow{\$} \mathsf{Initialize}$
3 : foreach $i \in [n]$
4 :     $(i, S) \xleftarrow{\$} \mathcal{A}^{O_1,\ldots,O_n}(C_1, \ldots, C_n)$
5 : return $\mathsf{Finalize}(C_i, \mathsf{state}_i, S)$
```

Fig. 1. Left: A single instance of a game. Right: A game in the multi-instance setting whereby O_i we denote a copy of the oracle set for the i-th game, that is, the adversary runs on all challenges and has access to the oracles of all games at the same time.

2.3 Blind Signatures

In the following, we introduce Blind Signatures schemes by extending the description by Fischlin & Schröder [13]. We do so by making use of the notation introduced earlier for interactive executions between two algorithms.

We introduce the syntax and security definitions of blind signature schemes and partially blind signature schemes. A fully blind signature scheme is a special case of a partially blind signature scheme where there is only one tag info, the empty string. We will refer to schemes where the tag is always the empty string as blind signature schemes.

Definition 8 (Partially Blind Signature Scheme). *A μ-move partially blind signature scheme Σ consists of a tuple of efficient algorithms* $(\mathsf{KGen}, \langle\mathsf{Sign}, \mathsf{User}\rangle, \mathsf{Vf})$ *with the following behaviour. We denote by $\mu_s := \lceil \mu/2 \rceil$ the number of messages sent by the signer, and by $\mu_u := \lfloor \mu/2 \rfloor$ the number of messages sent by the user.*[1]

- KGen*: the randomised key generation algorithm takes as input parameters* pp*, and outputs a public key* pk *and a secret key* sk*.*
- $\langle\mathsf{Sign}, \mathsf{User}\rangle$*: the joint execution of algorithms* Sign *and* User *.*
  - Sign*: the randomised algorithm takes as input a secret key* sk *and a tag* info*. For $i \in [\mu_s]$, it executes*

    $$(\mathsf{st}_i^S, \mathsf{msg}_i) \xleftarrow{\$} \mathsf{Sign}_i(\mathsf{sk}, \mathsf{info}, \mathsf{st}_{i-1}^S, \mathsf{chl}_{i-\mu_s+\mu_u})$$

    and returns the state st_i^S *and the output* msg_i*. For $\mu_s = \mu_u + 1$, and $i = 1$ we define* $\mathsf{chl}_{i\mu_s+\mu_u} = \mathsf{chl}_0$ *to be the empty string. We further define* st_0^S *to be the empty string.*
  - User*: the randomised algorithm takes as input a public key* pk*, a message m, a tag* info*. $i \in [\mu_u + 1]$, it executes*

    $$(\mathsf{st}_i^U, \mathsf{chl}_i) \xleftarrow{\$} \mathsf{User}_i(\mathsf{pk}, \mathsf{info}, m, \mathsf{st}_{i-1}^U, \mathsf{msg}_{i+\mu_s-(\mu_u+1)})$$

[1] W.l.o.g. we only consider schemes where the last message is sent by the signer, as the user can generate a signature as soon as he has received the last message from the signer.

and returns the state st_i^U *and the output* chl_i*, where for* User_{μ_u+1}*, the output is a signature* σ *or* $\perp$ *which are not sent to the signer. We further define* msg_0 *in the case that it occurs to be the empty string. We further define* st_0^U *to be the empty string.*

- *The joint execution generates the final output* $\perp$ *or a signature* σ *for* User *and some possibly empty final output* λ *for* Sign.

$$(\lambda, \sigma \vee \perp) \xleftarrow{\$} \langle \mathsf{Sign}(\mathsf{sk}, \mathsf{info}), \mathsf{User}(\mathsf{pk}, \mathsf{info}, m)\rangle$$

– Vf*: the deterministic algorithm takes as input a public key* pk*, a signature* σ*, and a message* m *and a tag* info*. The algorithm outputs either 1 indicating valid or 0 indicating not valid.*

Definition 9 (Correctness). *We say that a partially blind signature scheme* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User}\rangle, \mathsf{Vf})$ *is correct if for all* pp*, for all* $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$*, for all messages* m*, all tags* info *and all signatures* σ *outputted by* User *in the joint execution of* $\mathsf{Sign}(\mathsf{sk}, \mathsf{info})$ *and* $\mathsf{User}(\mathsf{pk}, \mathsf{info}, m)$ *we have* $\mathsf{Vf}(\mathsf{pk}, \sigma, m, \mathsf{info}) = 1$*.*

We consider the following notion of unforgeability for (partially) blind signature schemes:

Definition 10 (One-More-Unforgeability (OM-UF)). *For a blind signature scheme* Σ *and an adversary* $\mathcal{A}$ *we define the One-More-Unforgeability experiment as shown in fig. 2.*

```
$\mathsf{OM\text{-}UF}^{\Sigma}_{\mathcal{A},\ell}(\mathsf{pp})$
$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$
$((m_1, \sigma_1), \ldots, (m_{\ell+1}, \sigma_{\ell+1})) \xleftarrow{\$} \mathcal{A}^{\langle \mathsf{Sign}(\mathsf{sk}), \cdot \rangle^{\infty}}(\mathsf{pk})$
return 1 if
    $m_i \neq m_j$ for $1 \leq i < j \leq \ell + 1$ and
    $\mathsf{Vf}(\mathsf{pk}, m_i, \sigma_i) = 1$ foreach $i \in \{1, \cdots, \ell + 1\}$ and
    $\mathsf{Sign}$ has completed at most $\ell$ executions sessions
```

Fig. 2. One-More-Unforgeability game for an adversary $\mathcal{A}$ on a blind signature scheme Σ.

We denote the advantage of $\mathcal{A}$ *in the* $\mathsf{OM\text{-}UF}$ *game by*

$$\mathsf{Adv}^{\mathsf{OM\text{-}UF}}_{\Sigma,\mathcal{A},\ell} = \Pr\left[\mathsf{OM\text{-}UF}^{\Sigma}_{\mathcal{A},\ell} = 1\right]$$

and we say that Σ *is* (t, ε)*-*ℓ*-one-more-unforgeable if for any adversary* $\mathcal{A}$ *that runs in time at most* t*,* $\mathsf{Adv}^{\mathsf{OM\text{-}UF}}_{\Sigma,\mathcal{A},\ell} \leq \varepsilon$*.*

Another security property of (partially) blind signature schemes is (partial) blindness. As the focus of this work is unforgeability, we give the definition of (partial) blindness in appendix A.

3 Blind Signatures in the Multi-Signer Setting (MSS)

In this section, we will discuss the two definitions for the security of Blind Signatures in the *Multi-Signer Setting* as well as the translation of the notion of *Key Substitution* attacks [19].

3.1 Blind Signatures Key Substitution

Key Substitution entails appropriating a signature σ for a message m generated by a legitimate entity with the corresponding public key pk and making it appear to be the signature of another entity with the public key $\overline{\mathsf{pk}}$. An adversary could claim to be an entity with the public key $\overline{\mathsf{pk}}$ and pass off the signature σ as its own. We refer to such an attack as *Blind Signature Key Substitution Attack*, or simply *Key Substitution* when the context is implied. We adopt the definition of a similar attack against digital signatures from [19] as follows.

Definition 11 (Key Substitution Attacks for Blind Signatures). *For a Blind Signature Scheme Σ we define the game* KeySub *as described in fig. 3. We*

```
$\mathsf{KeySub}^{\Sigma}_{\mathcal{A}}(\mathsf{pp}, n)$
1 : foreach $i \in [n]$
2 :     $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(\mathsf{pp})$
3 : $(\mathsf{pk}^*, \overline{\mathsf{pk}}, m, \sigma) \leftarrow \mathcal{A}^{\langle \mathsf{Sign}(\mathsf{sk}_{i,i\in[n]}), \cdot \rangle^{\infty}}(\mathsf{pk}_{i,i\in[n]})$
4 : return $(\mathsf{pk}^* = \mathsf{pk}_i$ for some $i \in [n] \wedge \mathsf{pk}^* \neq \overline{\mathsf{pk}} \wedge \mathsf{Vf}(\mathsf{pk}^*, m, \sigma) \wedge \mathsf{Vf}(\overline{\mathsf{pk}}, m, \sigma))$
```

Fig. 3. Key Substitution game for an adversary $\mathcal{A}$ and a blind signature scheme Σ.

denote the advantage of an adversary $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{KeySub}}_{\Sigma,\mathcal{A},\mathsf{pp},n} = \Pr\left[\mathsf{KeySub}^{\Sigma}_{\mathcal{A}}(\mathsf{pp}, n) = 1\right]$$

and say that Σ is (t, ε)-secure against Key Substitution *Attacks if for all adversaries $\mathcal{A}$ that run in time at most t, $\mathsf{Adv}^{\mathsf{KeySub}}_{\Sigma,\mathcal{A}} \leq \varepsilon$.*

Remark 2 (On the Notion of Key Substitution Attacks). We choose this notion as it is the strongest one in the sense that it makes the least restrictions on the adversary. Weaker notions include those where the adversary is required to output a secret key matching its chosen public key $\overline{\mathsf{pk}}$, bounds on the set of permitted public keys $\overline{\mathsf{pk}}$ (e.g. only accepting public keys in the support of KGen), or bounds on the number of signing sessions permitted.

3.2 One-More Unforgeability in MSS

We introduce an extension of *One-More Unforgeablity* (definition 10), a well-accepted notion for blind signatures, for the *Multi-Signer Setting.* This is necessary to adequately account for the capabilities of an adversary in the real world. Indeed, doing so is critical as attacks on applications of blind signature schemes involving multiple entities can have damaging consequences.

Definition 12 (Strong-Multi-Signer-One-More-Unforgeability (sMSS-OM-UF)). *For a blind signature scheme Σ and an adversary $\mathcal{A}$, we define the Strong-Multi-Signer-One-More-Unforgeability game as*

$$\mathsf{sMSS\text{-}OM\text{-}UF}^{\Sigma}_{\mathcal{A},n,\ell} = \exists\text{-}n\text{-}\mathsf{OM\text{-}UF}^{\Sigma}_{\mathcal{A},\ell}$$

(see definitions 7 and 10) and say that Σ is (t, ε)-secure against sMSS-OM-UF *if for all adversaries that run in time t, it holds that the advantage*

$$\mathsf{Adv}^{\mathsf{sMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n,\ell} \leq \varepsilon.$$

For completeness, we provide pseudocode of the sMSS-OM-UF game in fig. 4

```
$\mathsf{sMSS\text{-}OM\text{-}UF}^{\Sigma}_{\mathcal{A}}(\mathsf{pp}, n)$
1 : foreach $i \in [n]$
2 :     $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(\mathsf{pp})$
3 : $(i^*, (m_j, \sigma_j)_{j\in[\ell+1]})) \leftarrow \mathcal{A}^{\langle \mathsf{Sign}(\mathsf{sk}_{i,i\in[n]}),\cdot\rangle^{\infty}}(\mathsf{pk}_{i,i\in[n]})$
4 : return 1 if
5 :     $m_j \neq m_{j'}$ foreach $j, j' \in [\ell + 1]$
6 :     and $\mathsf{Vf}(\mathsf{pk}_{i^*}, m_j, \sigma_{i,j}) = 1\, j \in [\ell + 1]$and
7 :     $\mathsf{Sign}(\mathsf{sk}_{i^*})$ has completed at most $\ell$ execution sessions
```

Fig. 4. Pseudocode for strong multi-signer one-more-unforgeability game.

We further define a weaker version of the above security notion that allows a tight security reduction for specific blind signature schemes.

Definition 13 (Weak-Multi-Signer-One-More-Unforgeability (wMSS-OM-UF)). *For a blind signature scheme Σ, and any adversary $\mathcal{A}$, we define the weak-multi-signer one-more-unforgeability game in fig. 5. whereby δ_{i,i^*} denotes* 1 *in the case that $i = i^*$ and* 0 *otherwise.*

We denote by

$$\mathsf{Adv}^{\mathsf{wMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n} := \Pr[\mathsf{wMSS\text{-}OM\text{-}UF}^{\Sigma}_{\mathcal{A}}(\mathsf{pp}, n) = 1]$$

```
$\mathsf{wMSS\text{-}OM\text{-}UF}^{\Sigma}_{\mathcal{A}}(\mathsf{pp}, n)$
1 : foreach $i \in [n]$
2 :    $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(\mathsf{pp})$
3 : $\big((m_{i^*,j}, \sigma_{i^*,j})_{j \in [\ell_{i^*}+1]}\big), (m_{i,j}, \sigma_{i,j})_{i \in [n] \setminus i^*, j \in [\ell_i]}\big)\big) \leftarrow \mathcal{A}^{\langle \mathsf{Sign}(\mathsf{sk}_{i,i \in [n]}), \cdot \rangle^{\infty}}(\mathsf{pk}_{i,i \in [n]})$
4 : return $1$ if
5 :    $m_{i,j} \neq m_{i',j'}$ foreach $i, i' \in [n], j \in [\ell_i + \delta_{i,i^*}], j' \in [\ell_{i'} + \delta_{i,i^*}], (i,j) \neq (i',j')$
6 :    and $\mathsf{Vf}(\mathsf{pk}_i, m_{i,j}, \sigma_{i,j}) = 1$ foreach $i \in [n], j \in [\ell_i + \delta_{i,i^*}]$ and
7 :    $\mathsf{Sign}(\mathsf{sk}_i)$foreach $i \in [n]$ has completed at most $\ell_i$ execution sessions
```

Fig. 5. Pseudocode for weak multi-signer one-more-unforgeability game.

and we say that Σ is (t, ε)-secure against One-More-Unforgeability in the weak Multi-Signer-Setting if for all adversaries $\mathcal{A}$ that run in time at most t it holds that

$$\mathsf{Adv}^{\mathsf{wMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n} \leq \varepsilon.$$

Remark 3 (On the weak vs. strong notion). Unlike some other security games, the OM-UF game can be "lost" in two different, not trivially equivalent ways. This is because an adversary may be able to still output as many signatures as it requested when it fails, or it may fail and not even be able to output as many signatures as requested. This is reflected in our two different notions of MSS-OM-UF, as in both cases, the adversary only needs to "win" one game. However, in the strong setting, it can lose all other games completely and output no signatures, whereas, in the weak setting, it still needs to output "enough" signatures in each game.

4 Generic Analysis of Blind Signatures in the Multi-Signer-Setting

In the following, we describe how Blind Signature schemes can be generically proven secure against the attack scenarios from the previous section.

4.1 Modifications against Key Substitution

We present two different modifications for blind signature schemes that aim to negate the probability of an attacker carrying out a *Key Substitution* attack in the presence of multiple signers. The main intuition behind both modifications is to make the signer's public key part of the message for which a signature is obtained so that a signature with respect to one public key is not a valid signature with respect to another public key.

Modified Verification Algorithm In [19], Menezes and Smart proposed a generic modification of digital signature schemes to counter the *Key Substitution* attack by prepending messages with the signer's public key before signing. In the case of blind signatures, a signer has no control over the messages he signs; hence, no way to prepend messages with his public key. Nevertheless, we adopt this modification for blind signatures by letting the user do the prepending and modifying the verification algorithm to check the format of the message.

Let $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ be a blind signature scheme. We define a blind signature scheme $\Sigma_1 = (\mathsf{KGen}_1, \langle \mathsf{Sign}_1, \mathsf{User}_1 \rangle, \mathsf{Vf}_1)$.

- $\mathsf{KGen}_1(\mathsf{pp}) := \mathsf{KGen}(\mathsf{pp})$
- $\langle \mathsf{Sign}_1(\mathsf{sk}), \mathsf{User}_1(\mathsf{pk}, m) \rangle := \langle \mathsf{Sign}(\mathsf{sk}), \mathsf{User}(\mathsf{pk}, \mathsf{pk}||m) \rangle$
- $\mathsf{Vf}_1(\mathsf{pk}, m, \sigma) := \mathsf{Vf}(\mathsf{pk}, \mathsf{pk}||m, \sigma)$

Theorem 1. *If a blind signature scheme* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *is secure against a one-more-forgery adversary in the Single-Signer Setting, then* $\Sigma_1 = (\mathsf{KGen}_1, \langle \mathsf{Sign}_1, \mathsf{User}_1 \rangle, \mathsf{Vf}_1)$ *is secure against a* Key Substitution *adversary in the Multi-Signer Setting.*

Proof. Let $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ be a one-more-forgery secure blind signature scheme in the *Single-Signer Setting.* Σ_1 is secure against *Key Substitution* attacks since none of the message-signature pairs that are valid with respect to a public key pk can be valid with respect to another public key $\overline{\mathsf{pk}}$. This is because the public key $\mathsf{pk} \neq \overline{\mathsf{pk}}$ is part of the message and is checked by the verification algorithm. We remind that a *Key Substitution* attacker of a blind signature scheme can only succeed if a message-signature pair is valid with respect to two different public keys. $\square$

The properties of our generically modified scheme are similar to those of restrictive blind signature schemes. Restrictive blind signatures, introduced by Brands [9], allow a recipient to obtain a blind signature on a message that is unknown to the signer, but the choice of message is restricted and must conform to certain rules.

For Partially Blind Schemes A partially blind signature scheme $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ can be modified to $\Sigma_2 = (\mathsf{KGen}_2, \langle \mathsf{Sign}_2, \mathsf{User}_2 \rangle, \mathsf{Vf}_2)$ to become secure against *Key Substitution* attacks by simply making the signer's public key part of the tag. The signing algorithm Sign_2 is identical to Sign, except that it checks that the tag contains the correct public key before signing and aborts if it does not. This is possible because the tag is not blinded.

- $\mathsf{KGen}_2(\mathsf{pp}) := \mathsf{KGen}(\mathsf{pp})$
- $\langle \mathsf{Sign}_2(\mathsf{sk}, \mathsf{info}), \mathsf{User}_2(\mathsf{pk}, \mathsf{info}, m) \rangle := \langle \mathsf{Sign}(\mathsf{sk}, \mathsf{pk}||\mathsf{info}), \mathsf{User}(\mathsf{pk}, \mathsf{pk}||\mathsf{info}, m) \rangle$
- $\mathsf{Vf}_2(\mathsf{pk}, m, \sigma, \mathsf{info}) := \mathsf{Vf}(\mathsf{pk}, m, \sigma, \mathsf{pk}||\mathsf{info})$

Theorem 2. *If a partially blind signature scheme* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *is secure against a one-more forgery adversary in the Single-Signer Setting, then* $\Sigma_2 = (\mathsf{KGen}_2, \langle \mathsf{Sign}_2, \mathsf{User}_2 \rangle, \mathsf{Vf}_2)$ *is secure against a* Key Substitution *adversary in the Multi-Signer Setting.*

Proof. Let $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ be a one-more-forgery secure partially blind signature scheme in the *Single-Signer Setting.* Σ_2 is secure against *Key Substitution* attacks since none of the (message, signature, tag) triples that are valid with respect to a public key pk can be valid with respect to another public key $\overline{\mathsf{pk}}$. This is because the public key $\mathsf{pk} \neq \overline{\mathsf{pk}}$ is part of the tag and is checked by the verification algorithm. The signer only creates signatures that are valid with respect to tags containing the public key pk. □

4.2 Lossy Reduction for One-More Unforgeability

In the following, we compare the advantage of a one-more forgery adversary in *Single-Signer Setting* to that of a one-more forgery adversary in the *Multi-Signer Setting.*

For any blind signature scheme, the following theorem states that the advantage of a one-more-forgery adversary in the *Multi-Signer Setting* can be upper bounded by a function of the advantage of a one-more forgery adversary with comparable resources in the *Single-Signer Setting.* The factor in the upper bound is polynomial in the number of signers in the setting and is parameterised by the maximum number of completed signing sessions that may be executed with each of the signing oracles. This is similar to existential unforgeability generic reductions for digital signatures in [16], and [14].

Theorem 3. *Let n, ℓ be integers and Σ a blind signature scheme. It holds that*

$$\mathsf{Adv}^{\mathsf{sMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n,\ell} \leq n \cdot \mathsf{Adv}^{\mathsf{OM\text{-}UF}}_{\Sigma,\mathcal{A},\ell}$$

where $\mathsf{Adv}^{\mathsf{sMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n,\ell}$ is defined in definition 12 and $\mathsf{Adv}^{\mathsf{OM\text{-}UF}}_{\Sigma,\mathcal{A},\ell}$ is defined in definition 10.

Proof. In the following, we prove the theorem from above. We assume that we have an algorithm $\mathcal{A}$ capable of generating One-More Forgery with probability ϵ_n in *Multi-Signer Setting* consisting of n signers.

We now describe an algorithm $\mathcal{B}_{\mathcal{A}}$ for solving the *Single-Signer* case by using the algorithm $\mathcal{A}$ as a subroutine. The input to $\mathcal{B}_{\mathcal{A}}$ is a public key pk. Further, access to the corresponding signing oracle $\mathsf{Sign}(\mathsf{sk})$ is given. Basically, $\mathcal{B}$ can simulate a game for $\mathcal{A}$ in the *Multi-Signer Setting* by generating multiple signing key pairs itself. A pseudo-code of $\mathcal{B}_{\mathcal{A}}$ is given in fig. 6.

We describe how $\mathcal{B}_{\mathcal{A}}$ answers the signing queries $\mathcal{A}$. If $i = i^*$, i.e. $\mathsf{sk}_{i^*} = \mathsf{sk}$, then $\mathcal{B}_{\mathcal{A}}$ can use the signing oracle $\mathsf{Sign}(\mathsf{sk})$. For $i \neq i^*$, i.e. $\mathsf{sk}_i \neq \mathsf{sk}$, $\mathcal{B}_{\mathcal{A}}$ can answer the signing queries by using the secret key sk_i. This works because all secret keys $\mathsf{sk}_i, i \neq i^*$ are known to $\mathcal{B}_{\mathcal{A}}$, so $\mathcal{B}_{\mathcal{A}}$ can perfectly simulate all signing oracles.

The algorithm $\mathcal{B}_{\mathcal{A}}$ will output one-more forgery with a probability ofϵ_n/n because there is a $1/n$ chance that $\mathsf{pk}_{i^*} = \mathsf{pk}$. Since $\epsilon_1 \geq \epsilon_n/n$ we get $\epsilon_n \leq n \cdot \epsilon_1$.

This proves the theorem above about a generic reduction from the *Multi-Signer* to the *Single-Signer Setting.*

□

```
$\mathcal{B}_{\mathcal{A}}(\mathsf{pk})$
1 : $i^* \xleftarrow{\$} [1, n]$
2 : $\mathsf{pk}_{i^*} \leftarrow \mathsf{pk}$
3 : foreach $i \in [1, n] \backslash [i^*]$
4 :     $(\mathsf{sk}_i, \mathsf{pk}_i) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
5 : $(\mathsf{pk}^*, m_t, \sigma_t)_{t \in [1, l+1]} \xleftarrow{\$} \mathcal{A}^{\langle \mathsf{Sign}(\mathsf{sk}_{i, i \in [1,n]}), \cdot \rangle^{\infty}}(\mathsf{pk}_{i, i \in [1,n]})$
6 : return $(\mathsf{pk}^*, m_t, \sigma_t)_{t \in [1, \ell+1]}$ if
7 :     $\mathsf{pk}^* = \mathsf{pk}_{i^*}$ and
8 :     $m_t \neq m_{t'}$ for $t, t' \in [1, \ell + 1], t \neq t'$ and
9 :     $\mathsf{Vf}(\mathsf{pk}^*, m_t, \sigma_t) = 1$ foreach $t \in [1, \ell + 1]$ and
10 :    $\mathsf{Sign}(\mathsf{sk}_{i^*})$ has completed at most $\ell$ execution sessions
11 : return fail
```

Fig. 6. Lossy generic reduction for s-MSS-OM-UF.

The following corollary allows us to conclude that, under certain assumptions, a blind signature scheme can be proven secure against a one-more forgery adversary in the *Multi-Signer Setting*. Since the relation between the advantages of $\mathcal{B}_{\mathcal{A}}$ and $\mathcal{A}$ is polynomial, we have the following:

Corollary 1. *Let* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *be a blind signature scheme that is polynomially secure against a one-more forgery adversary in the Single-Signer Setting. Then,* Σ *is also polynomially secure against a one-more forgery adversary in the Multi-Signer Setting.*

4.3 Trading the Reduction Loss for more Signing Sessions through Key Convertibility

We note that the reduction in section 4.2 loses a factor n where n is the number of instances/signers. When choosing parameters, e.g. for group sizes, for deploying blind signatures in the real world, a larger loss would mean that larger group parameters need to be chosen to achieve the same security guarantees.

Therefore, we are interested in a reduction that does not incur this loss. In the following, we present a way this reduction loss can be traded for an increased number of signing sessions in the *Single-Signer Setting*, as well as slightly weaker security guarantees in the *Multi-Signer Setting*.

This is possible when the schemes have certain properties. We call this general class of schemes *key convertible* blind signature schemes, similar to key convertible signature schemes introduced in [16].

We give our definition of a key converter for blind signature schemes below:

Definition 14 (Key Converter). *A* key converter C *for a blind signature scheme* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *is given by the following algorithms:*

– $\mathsf{C.Key}(\mathsf{pp},\mathsf{pk})$*: on input, a public key* pk *returns a public key* pk' *and a conversion key* $\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}$.
– $\mathsf{C.Conv}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'})$*: on input, a conversion key* $\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}$ *with access to a signing oracle* $\langle\mathsf{Sign}(\mathsf{sk}),\cdot\rangle^1$ *corresponding to the public key* pk *executes jointly with* User *to generate a signature for* User *and some possibly empty output* λ' *for* $\mathsf{C.Conv}$.

$$(\lambda',\sigma'\vee\bot)\xleftarrow{\$}\langle\mathsf{C.Conv}^{\langle\mathsf{Sign}(\mathsf{sk}),\cdot\rangle^1}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}),\mathsf{User}(\mathsf{pk}',m)\rangle$$

$\mathsf{C.Conv}$ *executes in the role of a user when interacting with* Sign *and in the role of a signer when interacting with* User*. Crucially,* $\mathsf{C.Conv}$ *only needs one completed joint execution session with* Sign *in order to generate a valid signature with respect to* pk' *for* User.
– $\mathsf{C.Rec}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'},\sigma',m)$*: on input, a conversion key* $\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}$ *and a signature-message pair* (σ',m) *valid in respect to* pk' *return a signature-message pair* (σ,m) *valid with respect to* pk.

The functionality of a key converter for a μ-move blind signature scheme is defined via the following requirements.

1. **Distribution of public keys**: for all $\mathsf{pp},\mathsf{pk}''$, $\{(\mathsf{pk}'',\mathsf{sk}'')\xleftarrow{\$}\mathsf{KGen}(\mathsf{pp})\}$, all $n\in\mathbb{N}$ the distributions
$$\{(\mathsf{pk}_i,\mathsf{sk}_i)\xleftarrow{\$}\mathsf{KGen}(\mathsf{pp}),i\in[n]:\mathsf{pk}_1,\dots,\mathsf{pk}_n\}$$
and
$$\{(\mathsf{pk}'_i,\mathsf{ck}_{\mathsf{pk}''\to\mathsf{pk}'_i})\xleftarrow{\$}\mathsf{C.Key}(\mathsf{pp},\mathsf{pk}''),i\in[n]:\mathsf{pk}'_1,\dots,\mathsf{pk}_n\}$$
must be identical.
2. **Distribution of interaction during signing session**: for all pp,pk with $(\mathsf{pk},\mathsf{sk})\xleftarrow{\$}\mathsf{KGen}(\mathsf{pp})$, $(\mathsf{pk}',\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'})\xleftarrow{\$}\mathsf{C.Key}(\mathsf{pp},\mathsf{pk})$ signing key sk' corresponding to pk', the following distributions of $\mathsf{C.Conv}$'s outputs in the role of a signer $\mathsf{msg}_{i\in[\mu_s]}$ throughout the interaction with a user, for any challenge $\mathsf{chl}_{i\in[\mu_u]}$ of the user's choice, defined as
$$\{(\mathsf{st}^s_i,\mathsf{msg}_i)\xleftarrow{\$}\mathsf{Sign}_i(\mathsf{sk}',\mathsf{st}^s_{i-1},\mathsf{chl}_i):\mathsf{msg}_{i\in[\mu_s]}\}$$
and
$$\{(\mathsf{st}^c_i,\mathsf{msg}'_i)\xleftarrow{\$}\mathsf{C.Conv}_i^{\langle\mathsf{Sign}(\mathsf{sk}),\cdot\rangle^1}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'},\mathsf{st}^c_{i-1},\mathsf{chl}_i):\mathsf{msg}'_{i\in[\mu_s]}\}$$
must be identical.
3. **Distribution of generated signatures**: for all public parameters pp, all pk s.t. $(\mathsf{pk},\mathsf{sk})\xleftarrow{\$}\mathsf{KGen}(\mathsf{pp})$, $(\mathsf{pk}',\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'})\xleftarrow{\$}\mathsf{C.Key}(\mathsf{pp},\mathsf{pk})$ signing key sk' corresponding to pk', and message m, the distributions
$$\{(\lambda,\sigma)\xleftarrow{\$}\langle\mathsf{Sign}(\mathsf{sk}'),\mathsf{User}(\mathsf{pk}',m)\rangle:\sigma\}$$
and
$$\{(\lambda',\sigma')\xleftarrow{\$}\langle\mathsf{C.Conv}^{\langle\mathsf{Sign}(\mathsf{sk}),\cdot\rangle^1}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}),\mathsf{User}(\mathsf{pk}',m)\rangle:\sigma'\}$$
must be identical.

4. **Recoverability of signatures**: for all parameters pp, public keys pk such that $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$, $(\mathsf{pk}', \mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}) \xleftarrow{\$} \mathsf{C.Key}(\mathsf{pp}, \mathsf{pk})$ messages m, signature σ' such that $\mathsf{Vf}(\mathsf{pk}', m, \sigma') = 1$ and $\sigma \leftarrow \mathsf{C.Rec}(\mathsf{ck}_{\mathsf{pk}\to\mathsf{pk}'}, \sigma', m)$ it must hold that $\mathsf{Vf}(\mathsf{pk}, m, \sigma) = 1$

With the above definition of a key converter for blind signature schemes, we are now ready to define a key convertible blind signature scheme.

Definition 15. *A blind signature scheme* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *is said to be key convertible if an efficient probabilistic polynomial time key converter* $\mathsf{C} = (\mathsf{C.Key}, \mathsf{C.Conv}, \mathsf{C.Rec})$ *for* Σ *exists.*

The following theorem shows that a tight reduction from *Multi-Signer* to the *Single-Signer Setting* can be obtained for a key convertible blind signature scheme.

Theorem 4 (The Knifing Lemma). *Let* $\Sigma = (\mathsf{KGen}, \langle \mathsf{Sign}, \mathsf{User} \rangle, \mathsf{Vf})$ *be a blind signature scheme that is key convertible. For any adversary* $\mathcal{A}$ *that runs in time at most* t *in the* wMSS-OM-UF *setting and makes* ℓ_i *signing queries for the i-th signer,* $i \in [n]$*, there exists an adversary* $\mathcal{B}$ *that also runs in time* t *such that:*

$$\mathsf{Adv}^{\mathsf{wMSS\text{-}OM\text{-}UF}}_{\Sigma,\mathcal{A},n} = \mathsf{Adv}^{\mathsf{OM\text{-}UF}}_{\Sigma,\mathcal{B},\sum_{i=1}^{n} \ell_i}$$

The following proof combines the technique used in [14] to prove a tight reduction for "Schnorr-like" digital signatures with the notion of key convertibility in [16].

Proof. Suppose we have an algorithm $\mathcal{A}$ which is able to win the wMSS-OM-UF security experiment with respect to Σ with a probability of ε. We construct $\mathcal{B}_{\mathcal{A}}$, which wins the OM-UF security experiment with respect to the same scheme, using $\mathcal{A}$ as a subroutine. The corresponding pseudo-code is provided in fig. 7.

Correctness of Simulated Public Keys. We show that the reduction $\mathcal{B}$ perfectly simulates the wMSS-OM-UF game to the adversary. We first show that the n public keys generated by $\mathcal{B}_{\mathcal{A}}$ are indeed identical to those generated by an honest wMSS-OM-UF challenger. This follows by using the first property of key convertible blind signature schemes n times.

Correctness of Signing Oracle Simulation. Algorithm $\mathcal{A}$ requires access to signing oracles for all public keys $\mathsf{pk}_i, i \in [1, n]$. We show how $\mathcal{B}_{\mathcal{A}}$ simulates the signing oracle $\mathsf{Sign}(\mathsf{sk}_i)$ corresponding to public key pk_i.

The reduction $\mathcal{B}_{\mathcal{A}}$ uses $\mathsf{C.Conv}(\mathsf{ck}_i)$ to simultaneously initiate a session with the signing oracle $\mathsf{Sign}(\mathsf{sk})$ in the role of a user and a session in the role of a signer with $\mathcal{A}$, which is running $\mathsf{User}(\mathsf{pk}_i, m)$, to generate a signature valid with respect to pk_i for $\mathcal{A}$. Using the fact that $(\mathsf{pk}_i, \mathsf{ck}_i) \xleftarrow{\$} \mathsf{C.Key}(\mathsf{pp}, \mathsf{pk})$, by second property of key convertible blind signature schemes, it must hold that the distributions

$$\{(\mathsf{st}^s_i, \mathsf{msg}_i) \xleftarrow{\$} \mathsf{Sign}_i(\mathsf{sk}', \mathsf{st}^s_{i-1}, \mathsf{chl}_i) : \mathsf{msg}_{i \in [\iota]}\}$$

```
$\mathcal{B}_{\mathcal{A}}(\mathsf{pp}, \mathsf{pk})$
1 : foreach $i \in [n]$
2 :   $(\mathsf{pk}_i, \mathsf{ck}_i) \leftarrow \mathsf{C.Key}(\mathsf{pp}, \mathsf{pk})$
3 : $(m_{i^*,j}, \sigma_{i^*,j})_{j \in [\ell_{i^*} + \delta_{i,i^*}]} \xleftarrow{\$} \mathcal{A}^{\langle \mathsf{C.Conv}^{\langle \mathsf{Sign(sk)}, \cdot \rangle^\infty}(\mathsf{ck}_{i, i \in [n]}), \cdot \rangle^\infty}(\mathsf{pk}_{i, i \in [n]})$
4 : return fail if not
5 :   $m_{i,j} \neq m_{i',j'}$ foreach $i, i' \in [n], j \in [\ell_i + \delta_{i,i^*}], j' \in [\ell_{i'} + \delta_{i',i^*}], (i,j) \neq (i',j')$
6 :   and $\mathsf{Vf}(\mathsf{pk}_i, m_{i,j}, \sigma_{i,j}) = 1$ foreach $i \in [n], j \in [\ell_i + \delta_{i,i^*}]$
7 :   and $\mathsf{Sign}(\mathsf{sk}_i)$ foreach $i \in [n]$ has completed at most $\ell_i$ execution sessions
8 : else return $(m_{i,j}, \mathsf{C.Rec}(\mathsf{ck}_i, \sigma_{i,j}))_{i \in [n], j \in [\ell_i + \delta_{i,i^*}]}$
```

Fig. 7. Reduction for weak multi-signer security of a key-convertible scheme. Here $\delta_{i,i^*} = 1$ if $i = i^*$ and 0 otherwise.

and

$$\{(\mathsf{st}_i^c, \mathsf{msg}_i') \xleftarrow{\$} \mathsf{C.Conv}_i^{\langle \mathsf{Sign(sk)}, \cdot \rangle^1}(\mathsf{ck}_{\mathsf{pk} \to \mathsf{pk}'}, \mathsf{st}_{i-1}^c, \mathsf{chl}_i) : \mathsf{msg}'_{i \in [\iota]}\}$$

must be identical. By using this property n times, it follows that $\mathcal{A}$ can not distinguish between outputs of signing oracles simulated by $\mathcal{B}_{\mathcal{A}}$ and outputs of signing oracles provided by an honest wMSS-OM-UF challenger.

Further, we show that all $\mathsf{C.Conv}(\mathsf{ck}_i)$ generate a valid signature with respect to pk_i for $\mathcal{A}$. By using the third property of key convertible blind signature schemes n times, it must hold that the distributions

$$\{(\lambda', \sigma') \xleftarrow{\$} \langle \mathsf{C.Conv}^{\langle \mathsf{Sign(sk)}, \cdot \rangle^1}(\mathsf{ck}_i), \mathsf{User}(\mathsf{pk}_i, m) \rangle : \sigma'\}$$

and

$$\{(\lambda, \sigma) \xleftarrow{\$} \langle \mathsf{Sign}(\mathsf{sk}_i), \mathsf{User}(\mathsf{pk}_i, m) \rangle : \sigma\}$$

are identical.

Hence, the above simulation of signing oracles is perfect.

Correctness of Output. We now show that $\mathcal{B}_{\mathcal{A}}$ outputs one more message-signature pair than the number of completed execution sessions with $\mathsf{Sign(sk)}$. Since at most ℓ_i sessions were completed for public key $\mathsf{pk}_i, i \in [1, n]$, as checked in line 7, and exactly one completed session with $\mathsf{Sign(sk)}$ is required to simulate one completed session with $\mathsf{Sign}(\mathsf{sk}_i)$ for $\mathcal{A}$, it follows that only at most $\sum_{i=1}^n \ell_i = \ell$ sessions are completed with $\mathsf{Sign(sk)}$ to simulate all signing oracles. Since $\mathcal{A}$ outputs $\ell_{i^*} + 1$ valid triples for one public key pk_{i^*} and ℓ_i for $i \neq i^*$, $\mathcal{B}_{\mathcal{A}}$ outputs $1 + \sum_{i=1}^n \ell_i = \ell + 1$ valid message-signature pairs with respect to pk by recovering signatures using C.Rec. Furthermore, as we required in wMSS-OM-UF (and in line 5) that all messages output are pairwise distinct (even across public keys), all $\ell + 1$ messages output by $\mathcal{B}_{\mathcal{A}}$ are pairwise distinct as well. This is indeed one more than the ℓ completed execution session with $\mathsf{Sign(sk)}$.

Clearly, the message-signature pairs in the list are valid with respect to pk (line 8) since the algorithm already checks in line 6 whether all triples outputted by $\mathcal{A}$ are valid. Due to the third property of key convertible blind signature schemes, it must hold that for $\sigma'_{jt} \leftarrow \mathsf{C.Rec}(\mathsf{ck}_i, \sigma_{jt})$, $\mathsf{Vf}(\mathsf{pk}, m_{jt}, \sigma'_{jt}) = 1$. Thus, all message-signature pairs outputted by the algorithm $\mathcal{B}_{\mathcal{A}}$ are valid with respect to pk.

Hence, $\mathcal{B}_{\mathcal{A}}$ produces one-more-forgery in the single-signer-setting.

□

5 Security of known Blind Signature Schemes in MSS

5.1 Blind BLS

bBLS is a secure blind signature scheme in the random oracle model given $\mathbb{G}$ is a GDH group [7]. It is based on the BLS digital signature scheme [8]. We recall $\mathsf{bBLS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ as follows: Let $\mathbb{G}$ be a GDH group. Let $\mathsf{H} : [\{0,1\}^* \to \mathbb{G}]$ be a hash function which maps arbitrary long strings to $\mathbb{G}$. Let $\mathsf{pp} = (p, g, \mathsf{H})$ be the public parameter containing the generator g, the prime group order p and the hash function H.

- KGen : On input $\mathsf{pp} = (p, g, \mathsf{H})$, KGen samples $x \xleftarrow{\$} \mathbb{Z}_p$ and computes $y \leftarrow g^x$. It sets $\mathsf{pk} \leftarrow y, \mathsf{sk} \leftarrow x$ and returns $(\mathsf{sk}, \mathsf{pk})$.
- Sign: On input a secret key sk, challenge $\overline{m}$, Sign computes $\overline{\sigma} \leftarrow (\overline{m})^x$ and returns the response $\overline{\sigma}$.
- User_1: On input a public key pk, a message $m \in \{0,1\}^*$, User_1 samples $r \xleftarrow{\$} \mathbb{Z}_p$ and computes $\overline{m} \leftarrow \mathsf{H}(m).g^r$. It returns the challenge $\overline{m}$, and the state $\mathsf{st}_u \leftarrow r$.
- User_2: On input a public key pk, a state $\mathsf{st}_u = r$ and a response $\overline{\sigma}$, User_2 computes $\sigma \leftarrow \overline{\sigma} \cdot y^{-r}$ and and returns σ.
- Vf: On input a public key pk, a message m and a signature σ, Vf checks $\log_g y \overset{?}{=} \log_{\mathsf{H}(m)} \sigma$ using the DDH solver. If so, it returns 1; otherwise, it returns 0.

A graphic of the scheme can be found in fig. 13 in appendix B.

Theorem 5. *Suppose* bBLS *is defined over a GDH group* $\mathbb{G}$ *with* $\mathsf{pp} = (p, g, \mathsf{H})$ *as a public parameter. Then, in the random oracle model,* bBLS *is secure against Key Substitution.*

Proof. Assume that an adversary $\mathcal{A}$ successfully outputs a public key $\mathsf{pk}' = y'$ different from $\mathsf{pk} = y$ such that (m, σ) is a valid message-signature pair with respect to both public keys. Since both signatures are valid, we have $\log_g y = \log_{\mathsf{H}(m)} \sigma$ and $\log_g y' = \log_{\mathsf{H}(m)} \sigma$. Since H is a random function, the probability of obtaining a collision is negligible. It follows that $\log_g y = \log_g y'$ and $y = y'$, which is a contradiction. □

Lemma 1. *The blind signature scheme* bBLS *is key convertible.*

Proof. We provide a key converter for bBLS in fig. 8. We prove that it fulfils the required properties below:

```
C.Key(pp, pk)
  1 : $\mathsf{ck} \xleftarrow{\$} \mathbb{Z}_p$
  2 : $\mathsf{pk}' \leftarrow \mathsf{pk}^{\mathsf{ck}}$
  3 : return $(\mathsf{pk}', \mathsf{ck})$

C.Rec(ck, σ')
  1 : $\alpha \leftarrow (\mathsf{ck})^{-1}$
  2 : $\sigma \leftarrow \sigma'^{\alpha}$
  3 : return $\sigma$

C.Conv_{u1}(m̄)
  1 : return $\overline{m}$

C.Conv_{s1}(ck, σ̄)
  1 : $\overline{\sigma}' \leftarrow \overline{\sigma}^{\mathsf{ck}}$
  2 : return $\overline{\sigma}'$
```

Fig. 8. A Key Converter for the Blind BLS Scheme

Distribution of public keys: On input a public parameter $\mathsf{pp} = (p, g, \mathsf{H})$ and a public key pk, $\mathsf{C.Key}$ implicitly sets the signing key sk' corresponding to pk' to $\mathsf{sk}' = r \cdot x$, where $x = \log_g(\mathsf{pk})$. Since ck is sampled uniformly at random from $\mathbb{Z}_p$ in each run of $\mathsf{C.Conv}$, for any n, an n-tuple of public keys generated by $\mathsf{C.Conv}$, with the same public key pk as input, is identically distributed to an n-tuple of public keys generated by KGen.

Distribution of interaction during signing session: Since $\overline{\sigma} = \overline{m}^x$ where $x = \log_g(y)$, we have that $\overline{\sigma'} = \overline{m}^{x \cdot ck}$ which is the unique valid response with respect to $\mathsf{sk}' = x \cdot ck$ and $\overline{m}$.

Distribution of generated signatures: Further, $\mathsf{C.Conv}$ generates the unique valid signature with respect to $\mathsf{pk}' = \mathsf{pk}^{\mathsf{ck}}$ for a message of the user's choice, as $H(m)^{\mathsf{sk} \cdot \mathsf{ck}} = H(m)^{\mathsf{sk}'}$.

Recoverability of signatures: Since $\overline{\sigma} = \overline{m}^x$ where $x = \log(\mathsf{pk})$, we have that

$$\sigma = \overline{m}^{x \cdot ck} \cdot g^{-x \cdot \mathsf{ck} \cdot r} = \mathsf{H}(m)^{x \cdot ck} \cdot g^{x \cdot \mathsf{ck} \cdot r - x \cdot \mathsf{ck} \cdot r} = \mathsf{H}(m)^{x \cdot ck} \qquad (1)$$

On input a conversion key ck and a valid signature σ' for a message m with respect to pk', $\mathsf{C.Rec}$ computes $\alpha \leftarrow (\mathsf{ck})^{-1}$ and returns σ^{α}.
We show that σ is a valid signature for message m with respect to pk. Since $\sigma' = m^{x \cdot ck}$, where $x = \log(y)$, $\sigma = m^{x \cdot \mathsf{ck} \cdot \alpha} = m^x$ is the unique valid signature for m with respect to pk.

□

5.2 Blind Schnorr Scheme (BSS)

We recall Schnorr's blind signature scheme BSS [24]: Let $\mathbb{G}$ be a group of order p with generator g, $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$ be a hash function and $\mathsf{pp} = (p, g, \mathsf{H})$ the public parameter.

– KGen : On input $\mathsf{pp} = (p, g, \mathsf{H})$, KGen samples $x \xleftarrow{\$} \mathbb{Z}_p$ and computes $y \leftarrow g^x$. It then sets $\mathsf{sk} \leftarrow x, \mathsf{pk} \leftarrow y$ and returns $(\mathsf{sk}, \mathsf{pk})$.

- Sign_1: On input a secret key sk, Sign_1 samples $r \xleftarrow{\$} \mathbb{Z}_p$ and computes $R \leftarrow g^r$. It then returns the commitment R and the state $\mathsf{st}_S = r$.
- Sign_2: On input a secret key sk, a state $\mathsf{st}_S = r$ and a challenge c, Sign_2 computes $s \leftarrow c \cdot \mathsf{sk} + r$ and returns the response s.
- User_1: On input a public key pk, a commitment R, and a message $m \in \{0,1\}^*$, User_1 does the following: It first samples $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$. Then, it computes $R' \leftarrow R \cdot g^\alpha \cdot \mathsf{pk}^\beta$ and $c' \leftarrow \mathsf{H}(R', m), c \leftarrow c' + \beta$. It returns the challenge c and the state $\mathsf{st}_U \leftarrow (R, c, \alpha, \beta, m)$.
- User_2: On input a public key pk, a state $\mathsf{st}_U = (R, c, \alpha, \beta, m)$, and a response s, User_2 first checks $g^s \overset{?}{=} R \cdot y^c$ and returns $\perp$ if not. Otherwise, it computes $R' \leftarrow R \cdot g^\alpha \cdot \mathsf{pk}^\beta$ and $s' \leftarrow s + \alpha$ and returns the signature (R', s').
- Vf On input a public key pk, a signature (R', s') and a message m, Vf computes $c' \leftarrow \mathsf{H}(R', m)$ and checks $g^{s'} \overset{?}{=} R' \cdot \mathsf{pk}^{c'}$. If so, it returns 1; otherwise, it returns 0.

A graphic of the scheme can be found in fig. 14 in appendix B.

Theorem 6. *Suppose that* BSS *is defined over a group* $\mathbb{G}$ *with* $\mathsf{pp} = (p, g, \mathsf{H})$ *as the public parameter. Then, in the random oracle model,* BSS *is secure against Key Substitution.*

Proof. Assume that an adversary $\mathcal{A}$ successfully outputs a public key $\overline{\mathsf{pk}} = \overline{y}$ different from $\mathsf{pk} = y$ such that $(m, (R', s'))$ is a valid message-signature pair with respect to both public keys. Since both signatures are valid, we have $g^{s'} = R' \cdot y^{c'}$, and $g^{s'} = R' \cdot \overline{y}^{c'}$, where $c' \leftarrow \mathsf{H}(R', m)$. Since H is a random function, the probability of obtaining a collision is negligible. It follows that $y^{c'} = \overline{y}^{c'}$ and $y = \overline{y}$, which is a contradiction. □

Lemma 2. *The blind signature scheme* BSS *is key convertible.*

Proof. We provide a key converter for BSS in fig. 9. We prove that it fulfils the required properties below:

Distribution of public keys: Since ck is sampled uniformly at random from $\mathbb{Z}_p$ in each run of $\mathsf{C.Conv}$, for any n, an n-tuple of public keys generated by $\mathsf{C.Conv}$, with the same public key pk as input, is identically distributed to an n-tuple of public keys generated by KGen.

Distribution of interaction during signing session: First, on input of the signer's commitment R, $\mathsf{C.Conv}$ forwards R to the user. Then, on input user's challenge c, it sets the state $\mathsf{st}^C \leftarrow c$ and forwards the c to the signer. Finally, on input a conversion key ck, a response s from the signer and a state $\mathsf{st}^C = c$, it computes a new response s'' and returns s'' to the user.

Since $R = g^r$ where r is randomly sampled by $\mathsf{Sign}(\mathsf{sk})$, $\mathsf{C.Conv}$'s first output is uniformly distributed and, thus, identical to the distribution of a commitment from $\mathsf{Sign}(\mathsf{sk}')$. We show that since r is randomly sampled by $\mathsf{Sign}(\mathsf{sk})$, $\mathsf{C.Conv}$'s second output s'' is uniformly distributed and also the

```
C.Key(pp, pk)
1 : ck ←$ Z_p
2 : pk' ← pk · g^{ck}
3 : return (pk', ck)

C.Rec(ck, m, (R', s'''))
1 : c' ← H(R', m)
2 : s' ← s''' − c' · ck
3 : σ ← (R', s')
4 : return σ

C.Conv_{s1}(R)
1 : return R

C.Conv_{u1}(c)
1 : st^C ← c
2 : return (st^C, c)

C.Conv_{s2}(ck, st^C, s)
1 : c ← st^C
2 : s'' ← s + ck · c
3 : return s''
```

Fig. 9. A Key Converter for the Blind Schnorr Scheme

unique response to the challenge c and $\mathsf{C.Conv}$'s first output $R = g^r$. Thus, the distribution of both outputs by $\mathsf{C.Conv}$ is identical to that of $\mathsf{Sign}(\mathsf{sk}')$. Since $s = c \cdot x + r$ where $x = \log_g(\mathsf{pk})$ and , we have that

$$s'' = c \cdot x + r + c \cdot \mathsf{ck} = c \cdot (x + \mathsf{ck}) + rs'' = c \cdot \mathsf{sk}' + r \tag{2}$$

Distribution of generated signatures: Further, $\mathsf{C.Conv}$ produces a valid signature (R', S') for message m with respect to pk' of the user's choice. To see this, first, we show that the user's check $g^{s''} \overset{?}{=} R \cdot \mathsf{pk}'^{c}$ holds.

$$g^{c \cdot \mathsf{sk}' + r} = g^r \cdot g^{\mathsf{sk}' \cdot c} = g^{c \cdot \mathsf{sk}' + r} \tag{3}$$

Then, we show that $\mathsf{Vf}(\mathsf{pk}', (R', s'), m)$ returns 1, by showing that

$$\begin{aligned} g^{s'} &= R' \cdot \mathsf{pk}'^{c'} & (4)\\ g^{s + \mathsf{ck} \cdot c + \alpha} &= R \cdot g^{\alpha} \cdot \mathsf{pk}'^{\beta} \cdot \mathsf{pk}'^{c'} & (5)\\ g^{(c'+\beta) \cdot x + r + \mathsf{ck} \cdot (c'+\beta) + \alpha} &= g^r \cdot g^{\alpha} \cdot g^{(x+\mathsf{ck}) \cdot \beta} \cdot g^{(x+\mathsf{ck}) \cdot c'} & (6)\\ g^{(c'+\beta) \cdot (x+\mathsf{ck}) + r + \alpha} &= g^{(c'+\beta) \cdot (x+\mathsf{ck}) + r + \alpha} & (7) \end{aligned}$$

Recoverability of signatures: On input a conversion key ck, a message m and a valid signature (R', s''') with respect to pk', $\mathsf{C.Rec}$ computes and returns (R', s''').
We show that (R', s') is a valid signature for message m with respect to pk. Since $x = \log(\mathsf{pk}), s'' = c \cdot \mathsf{sk}' + r, c' = \mathsf{H}(R', m)$ we have

$$\begin{aligned} s' = s''' - c' \cdot \mathsf{ck} &= c \cdot (x + \mathsf{ck}) + r + \alpha - c' \cdot \mathsf{ck} & (8)\\ &= (c' + \beta) \cdot x + (c' + \beta) \cdot \mathsf{ck} + r + \alpha - c' \cdot \mathsf{ck} & (9)\\ &= c' \cdot x + \beta \cdot x + \beta \cdot \mathsf{ck} + r + \alpha & (10) \end{aligned}$$

We show that $g^{s'} \overset{?}{=} R' \cdot \mathsf{pk}^{c'}$ holds.

$$g^{s'} = R \cdot g^{\alpha} \cdot g^{(x+\mathsf{ck})\beta} \cdot g^{x \cdot c'} \tag{11}$$

$$g^{c' \cdot x + \beta \cdot x + \beta \cdot \mathsf{ck} + r + \alpha} = g^{r + \alpha + x \cdot \beta + \mathsf{ck} \cdot \beta + x \cdot c'} \tag{12}$$

□

5.3 Okamoto-Schnorr Scheme (OSS)

OSS [20] was proved to be a secure blind signature scheme by Pointcheval and Stern in [21]. We recall OSS as as follows: OSS uses a group $\mathbb{G}$ of prime order p and two generators g, h of $\mathbb{G}$. It also uses hash function $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p$. We let the public parameter be $\mathsf{pp} = (p, g, h, \mathsf{H})$.

- KGen: On input $\mathsf{pp} = (p, g, h, \mathsf{H})$, KGen samples $r \xleftarrow{\$} \mathbb{Z}_p, s \xleftarrow{\$} \mathbb{Z}_p$ and computes $y \leftarrow g^{-r} h^{-s}$. It sets $\mathsf{sk} \leftarrow (r, s), \mathsf{pk} \leftarrow y$ and returns $(\mathsf{sk}, \mathsf{pk})$.
- Sign_1: On input a secret key sk, Sign_1 samples $t \xleftarrow{\$} \mathbb{Z}_p, u \xleftarrow{\$} \mathbb{Z}_p$and returns the commitment $a \leftarrow g^t h^u$ and the state $\mathsf{st}_S = (t, u)$.
- Sign_2: On input a secret key sk a state $\mathsf{st}_S = (t, u)$ and a challenge e, Sign_2 computes $R \leftarrow t + er$ and $S \leftarrow u + es$, and returns the pair (R, S).
- User_1: On input a public key pk, a commitment a, and a message m, User_1 does the following: It first samples $\beta, \gamma, \delta \xleftarrow{\$} \mathbb{Z}_p$, and it then computes $\alpha \leftarrow a g^{\beta} h^{\gamma} y^{\delta}$, $\varepsilon \leftarrow \mathsf{H}(m, \alpha)$ and $e \leftarrow \varepsilon - \delta$. It returns the challenge e and the state $\mathsf{st}_U \leftarrow (a, e, \beta, \gamma, \delta, m)$.
- User_2: On input a public key, a state $\mathsf{st}_U = (a, e, \beta, \gamma, \delta, m)$ and a response (R, S), User_2 first checks $a \overset{?}{=} g^R \cdot h^S \cdot y^e$ and returns $\perp$ if not. Otherwise, it computes $\rho \leftarrow R + \beta$ and $\sigma \leftarrow S + \gamma$ and returns the signature (α, ρ, σ).
- Vf: On input a public key pk, a signature (α, ρ, σ) and a message m, Vf computes $\varepsilon \leftarrow \mathsf{H}(m, \alpha)$ and checks $\alpha \overset{?}{=} g^{\rho} h^{\sigma} y^{\varepsilon}$.

A graphic of the scheme can be found in fig. 15 in appendix B.

Theorem 7. *Suppose that* OSS *is defined with* $\mathsf{pp} = (p, g, h, \mathsf{H})$ *as the public parameter. Then, in the random oracle model,* OSS *is secure against Key Substitution.*

Proof. Assume that an adversary $\mathcal{A}$ successfully outputs a public key $\overline{\mathsf{pk}} = \overline{y}$ different from $\mathsf{pk} = y$ such that $(m, (\alpha, \rho, \sigma))$ is a valid message-signature pair with respect to both public keys. Since both signatures are valid, we have $\alpha = g^{\rho} \cdot h^{\sigma} \cdot \overline{y}^{\varepsilon}$ and $\alpha = g^{\rho} \cdot h^{\sigma} \cdot y^{\varepsilon}$ where $\varepsilon \leftarrow \mathsf{H}(m, \alpha)$. Since H is a random function, the probability of obtaining a collision is negligible. It follows that $\overline{y}^{\varepsilon} = y^{\varepsilon}$ and $\overline{y} = y$, which is a contradiction. □

Lemma 3. *The blind signature scheme* OSS *is key convertible.*

Proof. We provide a key converter for OSS in fig. 10. We prove that it fulfils the required properties below:

```
C.Key(pp, pk)
1 : $\mathsf{ck}_r \xleftarrow{\$} \mathbb{Z}_p$
2 : $\mathsf{ck}_s \xleftarrow{\$} \mathbb{Z}_p$
3 : $\mathsf{pk}' \leftarrow \mathsf{pk} \cdot g^{-\mathsf{ck}_r} \cdot h^{-\mathsf{ck}_s}$
4 : return $(\mathsf{pk}', (\mathsf{ck}_r, \mathsf{ck}_s))$

C.Rec((ck_r, ck_s), m, (α', ρ', σ'))
1 : $\varepsilon \leftarrow \mathsf{H}(m, \alpha')$
2 : $\alpha \leftarrow \alpha'$
3 : $\rho \leftarrow \rho' - \varepsilon \cdot \mathsf{ck}_r$
4 : $\sigma \leftarrow \sigma' - \varepsilon \cdot \mathsf{ck}_s$
5 : return $(\alpha, \rho, \sigma)$

C.Conv_{s1}(a)
1 : return $a$

C.Conv_{u1}(e)
1 : $\mathsf{st}^C \leftarrow e$
2 : return $(\mathsf{st}^C, e)$

C.Conv_{s2}((ck_r, ck_s), st^C, (R, S))
1 : $e \leftarrow \mathsf{st}^C$
2 : $R' \leftarrow R + \mathsf{ck}_r \cdot e$
3 : $S' \leftarrow S + \mathsf{ck}_s \cdot e$
4 : return $(R', S')$
```

Fig. 10. A Key Converter for the Okamoto-Schnorr Scheme

Distribution of public keys: On input a public parameter $\mathsf{pp} = (p, g, h, \mathsf{H})$ and a public key pk, $\mathsf{C.Key}$ implicitly sets the signing key $\mathsf{sk}' = (r', s')$ corresponding to pk' to $r' = r + \mathsf{ck}_r, \mathsf{sk}'_s = s + \mathsf{ck}_s$, where $\mathsf{pk} = g^{-r} \cdot h^{-s}$. Since ck_r and ck_s are sampled uniformly at random from $\mathbb{Z}_p$ in each run of $\mathsf{C.Conv}$, for any n, an n-tuple of public keys generated by $\mathsf{C.Conv}$, with the same public key pk as input, is identically distributed to an n-tuple of public keys generated by KGen.

Distribution of interaction during the signing sessions: First, the algorithm forwards the signer's commitment a to the user. Then, it forwards the user's challenge e to the signer and sets the state $\mathsf{st}^C \leftarrow e$. Finally, on input a conversion key $(\mathsf{ck}_r, \mathsf{ck}_s)$, a response (R, S) from the signer and a state $\mathsf{st}^C = e$, it computes and returns a new response (R', S') to the user.
Since $a = g^t \cdot h^u$ where t and u are randomly sampled by $\mathsf{Sign}(\mathsf{sk})$, $\mathsf{C.Conv}$'s first output is uniformly distributed and, thus, identical to the distribution of a commitment from $\mathsf{Sign}(\mathsf{sk}')$. We show that since t and u are randomly sampled by $\mathsf{Sign}(\mathsf{sk})$, $\mathsf{C.Conv}$'s second output (R', S') is uniformly distributed and also the unique response to the challenge e and $\mathsf{C.Conv}$'s first output a. Thus, the distribution of both outputs by $\mathsf{C.Conv}$ is identical to that of $\mathsf{Sign}(\mathsf{sk}')$.

Distribution of generated signatures: Since $R = t + e \cdot r$ where $\mathsf{pk} = g^{-r} \cdot h^{-s}$, we have that

$$R' = t + e \cdot r + e \cdot \mathsf{ck}_r = t + e \cdot r' \tag{13}$$

Since $S = u + e \cdot s$ where $\mathsf{pk} = g^{-r} \cdot h^{-s}$, we have that

$$S' = u + e \cdot s + e \cdot \mathsf{ck}_s = u + e \cdot s' \tag{14}$$

Distribution of generated signatures: Furthermore, $\mathsf{C.Conv}$ produces a valid signature with respect to pk' for message m of the user's choice. First, we show that the user's check on R' and $S', a \overset{?}{=} g^{R'} \cdot h^{S'} \cdot y^e$ holds.

$$g^t \cdot h^u = g^{t+e\cdot r'} \cdot h^{u+e\cdot s'} \cdot (g^{-r'} \cdot h^{-s'})^e \tag{15}$$
$$= g^t \cdot h^u \cdot g^{e\cdot r'-e\cdot r'} \cdot h^{e\cdot s'-e\cdot s'} = g^t \cdot h^u \tag{16}$$

We show that $\mathsf{Vf}(\mathsf{pk}', (\alpha, \rho, \sigma), m)$ returns 1 by showing that

$$\alpha = g^\rho \cdot h^\sigma \cdot \mathsf{pk}'^\varepsilon \tag{17}$$
$$a \cdot g^\beta \cdot h^\gamma \cdot \mathsf{pk}'^\delta = g^{R'+\beta} \cdot h^{S'+\gamma} \cdot \mathsf{pk}'^{(e+\delta)} \tag{18}$$
$$= a \cdot g^\beta \cdot h^\gamma \cdot \mathsf{pk}'^\delta \tag{19}$$

Recoverability of Signatures: On input a conversion key $(\mathsf{ck}_r, \mathsf{ck}_s)$, a message m and a valid signature $(\alpha', \rho', \sigma')$ with respect to pk', the algorithm produces valid signature (α, ρ, σ) with respect to pk and m.
Since $\mathsf{pk} = y = g^{-r} \cdot h^{-s}, s'' = c \cdot \mathsf{sk}' + r, c' = \mathsf{H}(R', m)$, we have

$$\rho = \rho' - \varepsilon \cdot \mathsf{ck}_r = R' + \beta - \varepsilon \cdot \mathsf{ck}_r = t + e \cdot r' + \beta - \varepsilon \cdot \mathsf{ck}_r \tag{20}$$
$$= t + e \cdot (r + \mathsf{ck}_r) + \beta - \varepsilon \cdot \mathsf{ck}_r \tag{21}$$
$$= t + (\varepsilon - \delta) \cdot r + (\varepsilon - \delta) \cdot \mathsf{ck}_r + \beta - \varepsilon \cdot \mathsf{ck}_r = (t - \delta \cdot \mathsf{ck}_r) + e \cdot r + \beta \tag{22}$$

$$\sigma = \sigma' - \varepsilon \cdot \mathsf{ck}_s = S' + \gamma - \varepsilon \cdot \mathsf{ck}_s \tag{23}$$
$$= u + e \cdot s' + \gamma - \varepsilon \cdot \mathsf{ck}_s = u + e \cdot (s + \mathsf{ck}_s) + \gamma - \varepsilon \cdot \mathsf{ck}_s \tag{24}$$
$$= u + (\varepsilon - \delta) \cdot s + (\varepsilon - \delta) \cdot \mathsf{ck}_s + \gamma - \varepsilon \cdot \mathsf{ck}_s \tag{25}$$
$$= u + \varepsilon \cdot s - \delta \cdot s + \varepsilon \cdot \mathsf{ck}_s - \delta \cdot \mathsf{ck}_s + \gamma - \varepsilon \cdot \mathsf{ck}_s \tag{26}$$
$$= (u - \delta \cdot \mathsf{ck}_s) + e \cdot s + \gamma \tag{27}$$

We show that (α, ρ, σ) is a valid signature with respect to pk and m by showing that $\alpha = g^\rho \cdot h^\sigma \cdot \mathsf{pk}^\varepsilon$.

$$\alpha = a \cdot g^\beta \cdot h^\gamma \cdot \mathsf{pk}'^\delta = g^{t+\beta+(-r-\mathsf{ck}_r)\cdot\delta} \cdot h^{u+\gamma+(-s-\mathsf{ck}_s)\cdot\delta} \tag{28}$$

$$g^\rho \cdot h^\sigma \cdot \mathsf{pk}^\varepsilon = g^{(\rho'-\varepsilon\cdot\mathsf{ck}_r)} \cdot h^{(\sigma'-\varepsilon\cdot\mathsf{ck}_s)} \cdot \mathsf{pk}^{(\delta+e)} \tag{29}$$
$$= g^{(t-\delta\cdot\mathsf{ck}_r)+e\cdot r+\beta} \cdot h^{(u-\delta\cdot\mathsf{ck}_s)+e\cdot s+\gamma} \cdot (g^{-r} \cdot h^{-s})^{(\delta+e)} \tag{30}$$
$$= g^{t+\beta+(-r-\mathsf{ck}_r)\cdot\delta} \cdot h^{u+\gamma+(-s-\mathsf{ck}_s)\cdot\delta} \tag{31}$$

□

5.4 Tessaro-Zhu-GGM (TZG)

We recall the generically secure blind signature scheme from [25], which we denote by TZG: Let $\mathbb{G}$ be a prime-order cyclic group of order p with generator g, $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$ be a hash function and $\mathsf{pp} = (p, g, \mathsf{H})$ the public parameter.

- KGen : On input $\mathsf{pp} = (p, g, \mathsf{H})$, KGen samples $x \xleftarrow{\$} \mathbb{Z}_p$ and computes $X \leftarrow g^x$. It sets $\mathsf{sk} \leftarrow x, \mathsf{pk} \leftarrow X$ and returns $(\mathsf{sk}, \mathsf{pk})$.
- Sign_1: On input a secret key sk, Sign_1 samples $a \xleftarrow{\$} \mathbb{Z}_p, y \xleftarrow{\$} \mathbb{Z}_p^*$, computes $A \leftarrow g^a, \mathrm{Y} \leftarrow X^y$ and returns the state $\mathsf{st}^S \leftarrow (a, y, x)$ and the commitment (A, Y).
- Sign_2: On input a secret key sk, a state $\mathsf{st}^S = (a, y, x)$ and a challenge c, Sign_2 computes $s \leftarrow a + c \cdot y \cdot \mathsf{sk}$ and returns the response (s, y).
- User_1: On input a public key pk, a commitment (A, Y), and a message $m \in \{0,1\}^*$, User_1 does the following: It first samples $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p, \gamma \xleftarrow{\$} \mathbb{Z}_p^*$. Then, it computes $\mathrm{Y}' \leftarrow \mathrm{Y}^\gamma, A' \leftarrow g^{r_1} \cdot A^\gamma \cdot \mathrm{Y}'^{r_2}, c' \leftarrow \mathsf{H}(A'||\mathrm{Y}'||m)$, and $c \leftarrow c' + r_2$. It returns state $\mathsf{st}^U \leftarrow (c, c', r_1, \gamma, \mathsf{pk}, \mathrm{Y}, A)$ and challenge c.
- User_2: On input a public key pk, a state $\mathsf{st}^U = (c, c', r_1, \gamma, \mathsf{pk}, \mathrm{Y}, A)$, and a response (s, y), User_2 first checks whether $y \overset{?}{\neq} 0$ and $\mathrm{Y} \overset{?}{=} \mathsf{pk}^y$ and $g^s \overset{?}{=} A \cdot \mathrm{Y}^c$. It returns $\perp$ if not. Otherwise, it computes $s' \leftarrow \gamma \cdot s + r_1$ and $y' \leftarrow \gamma \cdot y$ and returns the signature (c', s', y').
- Vf: On input a public key pk, a signature (c', s', y') and a message m, Vf computes $\mathrm{Y} \leftarrow \mathsf{pk}^y, A \leftarrow g^{s'} \cdot \mathrm{Y}^{-c'}$ and checks whether $c' \overset{?}{=} \mathsf{H}(A||\mathrm{Y}||m)$. If so, it returns 1; otherwise, it returns 0.

A graphic of the scheme can be found in fig. 16 in appendix B.

Theorem 8. *Suppose that* TZG *is defined over a group* $\mathbb{G}$ *with* $\mathsf{pp} = (p, g, \mathsf{H})$ *as the public parameters. Then, in the random oracle model,* TZG *is secure against Key Substitution.*

Proof. Assume that an adversary $\mathcal{A}$ successfully outputs a public key $\overline{\mathsf{pk}} = \overline{X}$ different from $\mathsf{pk} = X$ such that $(m, (c', s', y'))$ is a valid message-signature pair with respect to both public keys. Since both signatures are valid, we have $c' = \mathsf{H}(g^{s'} \cdot (\mathsf{pk}^{y'})^{-c'}, \mathsf{pk}^{y'}, m)$ and $c' = \mathsf{H}(g^{s'} \cdot (\overline{\mathsf{pk}}^{y'})^{-c'}, \overline{\mathsf{pk}}^{y'}, m)$. Since H is a random function, the probability of obtaining a collision is negligible. It follows that $\mathsf{pk}^{y'} = \overline{\mathsf{pk}}^{y'}$ and $\mathsf{pk} = \overline{\mathsf{pk}}$, which is a contradiction. $\square$

Lemma 4. *The blind signature scheme* TZG *is key convertible.*

Proof. We provide a key converter for TZG in fig. 11. We prove that it fulfils the required properties below:

Distribution of public keys: For any n, an n-tuple of keys generated through key conversion is identically distributed to a fresh key as in a prime order group, exponentiation with a random value from $\mathbb{Z}_p$ yields a uniformly random group element. Thus, an n-tuple of converted public keys is a tuple of uniformly random group elements, just like an n-tuple of freshly generated public keys.

```
$\mathsf{C.Key}(\mathsf{pp}, \mathsf{pk})$
1 : $\mathsf{ck} \xleftarrow{\$} \mathbb{Z}_p$
2 : $\mathsf{pk}' \leftarrow \mathsf{pk}^{\mathsf{ck}}$
3 : return $(\mathsf{pk}', \mathsf{ck})$
```

```
$\mathsf{C.Rec}(\mathsf{ck}, m, (c', s', y'))$
1 : $c^* \leftarrow c'$
2 : $s^* \leftarrow s'$
3 : $y^* \leftarrow y' \cdot \mathsf{ck}$
4 : return $(c^*, s^*, y^*)$
```

```
$\mathsf{C.Conv}_{s1}(\mathsf{ck}, (A, \mathrm{Y}))$
1 : $\mathrm{Y}'' \leftarrow \mathrm{Y}^{\mathsf{ck}}$
2 : return $(A, \mathrm{Y}'')$
```

```
$\mathsf{C.Conv}_{u1}(\mathsf{ck}, c)$
1 : $c'' \leftarrow c \cdot \mathsf{ck}$
2 : return $c''$
```

```
$\mathsf{C.Conv}_{s2}((s, y))$
1 : return $(s, y)$
```

Fig. 11. A Key Converter for Tessaro-Zhu-GGM scheme

Distribution of interactions during the signing sessions: First, on input a conversion key ck and a signer's commitment (A, Y), $\mathsf{C.Conv}$ computes $\mathrm{Y}'' \leftarrow \mathrm{Y}^{\mathsf{ck}}$ and returns a new commitment (A, Y'') to the user. Then, on input ck and a user's challenge c, it computes $c'' \leftarrow \mathsf{ck} \cdot c$ and returns a new challenge c'' to the signer. Finally, it forwards the signer's response (s, y) to the user.

Since the commitment $(A = g^a, \mathrm{Y} = \mathsf{pk}^y)$ by $\mathsf{Sign}(\mathsf{sk})$ is uniformly distributed, where a and y are randomly sampled, $\mathsf{C.Conv}$'s first output $(g^a, \mathsf{pk}^{y \cdot \mathsf{ck}} = \mathsf{pk}'^{y})$ is also uniformly distributed and, thus, identical to the distribution of a commitment from $\mathsf{Sign}(\mathsf{sk}')$. We show that $\mathsf{C.Conv}$'s second output (s, y) is uniformly distributed as well as the unique response to a challenge c and $\mathsf{C.Conv}$'s first output, and also identical to the distribution of a response from $\mathsf{Sign}(\mathsf{sk}')$. Thus, the distribution of both outputs by $\mathsf{C.Conv}$ is identical to that of $\mathsf{Sign}(\mathsf{sk}')$.

Since a and y are randomly sampled by $\mathsf{Sign}(\mathsf{sk})$ and $s = a + c'' \cdot y \cdot x$ where $x = \log_g(\mathsf{pk})$, we have that

$$s = a + c'' \cdot y \cdot x = a + c \cdot \mathsf{ck} \cdot y \cdot x = a + c \cdot y \cdot \mathsf{sk}' \tag{32}$$

Distribution of generated signatures: Moreover, $\mathsf{C.Conv}$ produces a valid signature with respect to pk' for message m of the user's choice. First, we show that the user's checks on a signer's response

$$y \overset{?}{\neq} 0, \mathrm{Y}'' \overset{?}{=} \mathsf{pk}'^{y}, g^s \overset{?}{=} A \cdot \mathrm{Y}''^{c}$$

hold. Since $\mathsf{Sign}(\mathsf{sk})$ samples y form $\mathbb{Z}_p^*$, it follows that $y \neq 0$. Since $\mathrm{Y}'' = \mathsf{pk}^{y \cdot \mathsf{ck}}, x = \log(\mathsf{pk})$ and $s = a + c'' \cdot y \cdot x$

$$\mathrm{Y}'' = \mathsf{pk}^{\mathsf{ck} \cdot y} = \mathsf{pk}'^{y} \tag{33}$$

$$\begin{aligned} g^{s} &= A \cdot \mathrm{Y}''^{c} && (34)\\ g^{a+c''\cdot y\cdot x} &= g^{a} \cdot g^{x\cdot y\cdot \mathsf{ck}} && (35)\\ g^{a+c\cdot y\cdot \mathsf{sk}'} &= g^{a+c\cdot y\cdot \mathsf{sk}'} && (36)\end{aligned}$$

We now show that $\mathsf{Vf}(\mathsf{pk}', (c', s', y'), m)$ returns 1 by showing that

$$\mathrm{Y}' = \mathsf{pk}'^{y'}, A' = g^{s'} \cdot (\mathsf{pk}'^{y'})^{-c'}$$

thus

$$c' \stackrel{?}{=} \mathsf{H}(g^{s'} \cdot (\mathsf{pk}'^{y'})^{-c'}, \mathsf{pk}'^{y'}, m)$$

holds. Since $x = \log(\mathsf{pk}), \mathrm{Y}'' = \mathsf{pk}'^{y} = g^{x\cdot y\cdot \mathsf{ck}}$, we have that

$$\begin{aligned} \mathrm{Y}' &= \mathsf{pk}'^{y'} && (37)\\ \mathrm{Y}''^{\gamma} &= \mathsf{pk}'^{\gamma\cdot y} && (38)\\ g^{x\cdot y\cdot \mathsf{ck}\cdot\gamma} &= g^{x\cdot y\cdot \mathsf{ck}\cdot\gamma} && (39)\end{aligned}$$

and since $s = a + c'' \cdot y \cdot \mathsf{sk} = a + c \cdot y \cdot \mathsf{sk}'$, we have that

$$\begin{aligned} A' &= g^{s'} \cdot (\mathsf{pk}'^{y})^{-c'} && (40)\\ g^{r_1} \cdot A^{\gamma} \cdot \mathrm{Y}'^{r_2} &= g^{\gamma\cdot s+r_1} \cdot g^{-\mathsf{sk}'\cdot y\cdot c'} && (41)\\ g^{r_1} \cdot g^{a\cdot\gamma} \cdot \mathrm{Y}''^{\gamma\cdot r_2} &= g^{\gamma\cdot(a+c'\cdot y\cdot \mathsf{sk}')+r_1} \cdot g^{-\mathsf{sk}\cdot y\cdot c'} && (42)\\ g^{r_1} \cdot g^{a\cdot\gamma} \cdot g^{\mathsf{sk}\cdot y\cdot \mathsf{ck}\cdot\gamma\cdot r_2} &= g^{\gamma\cdot(a+\cdot(c'+r_2)\cdot y\cdot \mathsf{sk}')+r_1} \cdot g^{-\mathsf{sk}'\cdot\gamma\cdot y\cdot c'} && (43)\\ g^{r_1+a\cdot\gamma+\mathsf{sk}'\cdot y\cdot\gamma\cdot r_2} &= g^{r_1+a\cdot\gamma+\mathsf{sk}'\cdot y\cdot\gamma\cdot r_2} && (44)\end{aligned}$$

Recoverability of signatures: On input a conversion key ck, a message m and a valid signature (c', s', y') with respect to pk' and m, the algorithm produces a valid signature (c^*, s^*, y^*) with respect to pk and m.
We show that (c^*, s^*, y^*) is a valid signature with respect to pk, and m by showing that

$$c^* = \mathsf{H}(g^{s^*} \cdot (\mathsf{pk}^{y^*})^{-c^*} || \mathsf{pk}^{y^*} || m)$$

Since $c^* = c'$ where $c' = \mathsf{H}(A'||\mathrm{Y}'||m)$, we proceed by showing that $\mathrm{Y}' = \mathsf{pk}^{y*}$ and $A' = g^{s^*} \cdot (\mathsf{pk}^{y^*})^{-c^*}$ to conclude that the following holds.

$$\mathsf{H}(A'||\mathrm{Y}'||m) = \mathsf{H}(g^{s*} \cdot \mathrm{Y}^{-c^*} || \mathsf{pk}^{y*} || m)$$

Since $x = \log(\mathsf{pk}), \mathsf{sk}' = x \cdot \mathsf{ck}, s = a + c \cdot y \cdot \mathsf{sk}'$, we have

$$\begin{aligned} \mathrm{Y}' &= \mathsf{pk}^{y^*} && (45)\\ \mathsf{pk}'^{y\cdot\gamma} &= \mathsf{pk}^{y\cdot\gamma\cdot \mathsf{ck}} && (46)\\ g^{x\cdot \mathsf{ck}\cdot\gamma\cdot y} &= g^{x\cdot \mathsf{ck}\cdot\gamma\cdot y} && (47)\end{aligned}$$

$$
\begin{aligned}
A' &= g^{s^*} \cdot (\mathsf{pk}^{y*})^{-c^*} && (48)\\
g^{r_1} \cdot A^{\gamma} \cdot \mathrm{Y}'^{r_2} &= g^{s'} \cdot g^{-x \cdot \mathsf{ck} \cdot \gamma \cdot y \cdot c'} && (49)\\
g^{r_1} \cdot g^{a \cdot \gamma} \cdot \mathrm{Y}^{\gamma \cdot r_2} &= g^{\gamma \cdot s + r_1} \cdot g^{-x \cdot \mathsf{ck} \cdot \gamma \cdot y \cdot c'} && (50)\\
g^{r_1} \cdot g^{a \cdot \gamma} \cdot \mathsf{pk}'^{y \cdot \gamma \cdot r_2} &= g^{\gamma \cdot (a + c \cdot y \cdot x \cdot \mathsf{ck}) + r_1} \cdot g^{-x \cdot \mathsf{ck} \cdot \gamma \cdot y \cdot c'} && (51)\\
g^{r_1} \cdot g^{a \cdot \gamma} \cdot g^{\mathsf{sk}' \cdot y \cdot \gamma \cdot r_2} &= g^{\gamma \cdot (a + (c' + r_2) \cdot y \cdot x \cdot \mathsf{ck}) + r_1} \cdot g^{-x \cdot \mathsf{ck} \cdot \gamma \cdot y \cdot c'} && (52)\\
g^{r_1 + a \cdot \gamma + x \cdot \mathsf{ck} \cdot y \cdot \gamma \cdot r_2} &= g^{r_1 + a \cdot \gamma + x \cdot \mathsf{ck} \cdot y \cdot \gamma \cdot r_2} && (53)
\end{aligned}
$$

□

5.5 Tessaro-Zhu-AGM (TZA)

We recall the algebraically secure blind signature scheme from [25]: Let $\mathbb{G}$ be a prime-order cyclic group of order p with generator g, $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$ be a hash function and $\mathsf{pp} = (p, g, \mathsf{H})$ the public parameter.

- KGen : On input $\mathsf{pp} = (p, \mathbb{G}, g, \mathsf{H})$, KGen samples $x \xleftarrow{\$} \mathbb{Z}_p, Z \leftarrow \mathbb{G}$ and computes $X \leftarrow g^x$. It sets $\mathsf{sk} \leftarrow x, \mathsf{pk} \leftarrow (X, Z)$ and returns $(\mathsf{sk}, \mathsf{pk})$.
- Sign_1: On input a secret key sk, Sign_1 samples $a, t \xleftarrow{\$} \mathbb{Z}_p, y \xleftarrow{\$} \mathbb{Z}_p^*$, computes $A \leftarrow g^a, C \leftarrow g^t Z^y$ and returns the state $\mathsf{st}^S \leftarrow (a, y, t, x)$ and the commitment (A, C).
- Sign_2: On input a secret key sk, a state $\mathsf{st}^S = (a, y, x)$ and a challenge c, Sign_2 computes $s \leftarrow a + c \cdot y \cdot \mathsf{sk}$ and returns the response (s, y, t).
- User_1: On input a public key $(\mathsf{pk}_x, \mathsf{pk}_z)$, a commitment (A, C), and a message $m \in \{0,1\}^*$, User_1 does the following: It first samples $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$, $\gamma_1, \gamma_2 \xleftarrow{\$} \mathbb{Z}_p^*$. Then, it computes $C' \leftarrow C_1^{\gamma} \cdot g^{r_2}, A' \leftarrow g^{r_1} \cdot A^{\gamma_1/\gamma_2}, c' \leftarrow \mathsf{H}(A' || C' || m)$, and $c \leftarrow c' \cdot \gamma_2$. It returns state $\mathsf{st}^U \leftarrow (c, c', r_1, r_2, \gamma_1, \gamma_2, \mathsf{pk}_x, \mathsf{pk}_z, A, C)$ and challenge c.
- User_2: On input a public key $(\mathsf{pk}_x, \mathsf{pk}_z)$, a state $\mathsf{st}^U \leftarrow (c, c', r_1, r_2, \gamma_1, \gamma_2, \mathsf{pk}_x, \mathsf{pk}_z, A, C)$, and a response (s, y, t), User_2 first checks whether $y \overset{?}{\neq} 0$ and $C \overset{?}{=} g^t \cdot \mathsf{pk}_z^y$ and $g^s \overset{?}{=} A \cdot \mathsf{pk}_x^{c \cdot y}$. It returns $\perp$ if not. Otherwise, it computes $s' \leftarrow (\gamma_1/\gamma_2) \cdot s + r_1, y' \leftarrow \gamma_1 \cdot y, t' \leftarrow \gamma_1 \cdot t + r_2$ and returns the signature (c', s', y', t').
- Vf: On input a public key $(\mathsf{pk}_x, \mathsf{pk}_z)$, a signature (c', s', y', t') and a message m, Vf checks whether $c' \overset{?}{=} \mathsf{H}(g^{s'} \cdot \mathsf{pk}_x^{-c' \cdot y'} || g^{t'} \cdot \mathsf{pk}_z^{y'} || m)$. If so, it returns 1; otherwise, it returns 0.

A graphic of the scheme can be found in fig. 17 in appendix B.

Theorem 9. *Suppose that* TZA *is defined over a group* $\mathbb{G}$ *with* $\mathsf{pp} = (p, \mathbb{G}, g, \mathsf{H})$ *as the public parameters. Then, in the random oracle model,* TZA *is secure against Key Substitution.*

Proof. Assume that an adversary $\mathcal{A}$ successfully outputs a public key $\overline{\mathsf{pk}} = (\overline{X}, \overline{Z})$ different from $\mathsf{pk} = (X, Z)$ such that $(m, (c', s', y', t'))$ is a valid message-signature pair with respect to both public keys. Since both signatures are valid, we have $c' = \mathsf{H}(g^{s'} \cdot X^{-c' \cdot y'} || g^{t'} \cdot Z^{y'} || m)$ and $c' = \mathsf{H}(g^{s'} \cdot \overline{X}^{-c' \cdot y'} || g^{t'} \cdot \overline{Z}^{y'} || m)$ It follows that $g^{t'} \cdot Z^{y'} = g^{t'} \cdot \overline{Z}^{y'}$ and $Z = \overline{Z}$. Since H is a random function, the probability of obtaining a collision is negligible. It follows that $g^{s'} \cdot X^{-c' \cdot y'} = \mathsf{H}(g^{s'} \cdot \overline{X}^{-c' \cdot y'}$ and $X = \overline{X}$. From both follows $\mathsf{pk} = \overline{\mathsf{pk}}$, which is a contradiction. $\square$

Lemma 5. *The blind signature scheme* TZA *is key convertible.*

Proof. We provide a key converter for TZA in fig. 12. We prove that it fulfils the required properties below:

```
$\mathsf{C.Key}(\mathsf{pp}, (\mathsf{pk}_x, \mathsf{pk}_z))$
1 : $\mathsf{ck}_z, \mathsf{ck}_x \xleftarrow{\$} \mathbb{Z}_p$
2 : $\mathsf{pk}'_z \leftarrow \mathsf{pk}_z \cdot g^{\mathsf{ck}_z}$
3 : $\mathsf{pk}'_x \leftarrow \mathsf{pk}_x \cdot g^{ck_x}$
4 : **return** $((\mathsf{ck}_z, \mathsf{ck}_x), (\mathsf{pk}'_x, \mathsf{pk}'_z))$
```

```
$\mathsf{C.Rec}((\mathsf{ck}_x, \mathsf{ck}_z), m, (c', s', y', t'))$
1 : $c^* \leftarrow c'$
2 : $s^* \leftarrow s' - \mathsf{ck}_x \cdot c' \cdot y'$
3 : $y^* \leftarrow y'$
4 : $t^* \leftarrow t' + \mathsf{ck}_z \cdot y'$
5 : **return** $(c^*, s^*, y^*)$
```

```
$\mathsf{C.Conv}_{s1}((\mathsf{ck}_x, \mathsf{ck}_z), \mathsf{msg}_1)$
1 : **return** $\mathsf{msg}_1$
```

```
$\mathsf{C.Conv}_{u1}((\mathsf{ck}_x, \mathsf{ck}_z), c)$
1 : **return** $c$
```

```
$\mathsf{C.Conv}_{s2}((\mathsf{ck}_x, \mathsf{ck}_z), \mathsf{msg}_2, c)$
1 : $(s, y, t) \leftarrow \mathsf{msg}_2$
2 : $t'' \leftarrow t - \mathsf{ck}_z \cdot y$
3 : $s'' \leftarrow s - \mathsf{ck}_x \cdot c \cdot y$
4 : **return** $(s'', y, t'')$
```

Fig. 12. A Key Converter for the Tessaro-Zhu-AGM scheme

Distribution of public keys: For any n, an n-tuple of converted public keys is identically distributed to an n-tuple of freshly generated keys as we re-randomize both components of the public key using uniform values from $\mathbb{Z}_p$.

Distribution of interaction during the signing sessions: The first message (A, C) sent by the signer is identically distributed to that of a real signer as the values are forwarded and the original commitments are uniformly random group elements.
The second message output by the converter is identically distributed to that of a real signer. This is because c, y, the converted public key pk', as

well as the first message (A, C), the values s'', t'' are uniquely determined. The value y is chosen by the real signer and thus uniformly random, and the value c comes from the user. We show that the conversion algorithm outputs the uniquely determined values for s'', t'', that is $A = g^{s''} \cdot \mathsf{pk}_x'^{c\cdot y}$ and $C = g^{t''} \cdot \mathsf{pk}_z'^{y}$.
This is easy to see in the following:

$$\begin{aligned} A =& g^s \cdot \mathsf{pk}_x^{c\cdot y} && (54)\\ =& g^s \cdot (\mathsf{pk}_x' \cdot g^{-\mathsf{ck}_x})^{c\cdot y} && (55)\\ =& g^{s-\mathsf{ck}_x\cdot c\cdot y} \cdot \mathsf{pk}_x'^{c\cdot y} && (56)\\ =& g^{s''} \cdot \mathsf{pk}_x'^{c\cdot y} && (57)\end{aligned}$$

and

$$\begin{aligned} C =& g^t \cdot \mathsf{pk}_z^y = g^t \cdot (\mathsf{pk}_z' \cdot g^{-\mathsf{ck}_z})^y = g^t \cdot \mathsf{pk}_z'^y \cdot g^{-\mathsf{ck}_z \cdot y} && (58)\\ =& g^{t-\mathsf{ck}_z\cdot y} \cdot \mathsf{pk}_z'^y = g^{t''} \cdot \mathsf{pk}_z'^y && (59)\end{aligned}$$

Thus, the interactions during the signing sessions are identically distributed to those of a reals signer.

Distribution of generated signatures: To see that the signatures generated by the interaction of the user with the converter are valid signatures, we invoke the previous property as well as the correctness of the original scheme.

Recoverability of signatures: To see that the recovery of signatures works, we need to show that the recovered signature is valid with respect to the public key pk. We show that $\mathsf{H}(g^{s^*} \cdot \mathsf{pk}_x^{-c^*\cdot y^*} \| g^{t^*} \cdot \mathsf{pk}_z^{y^*} \| m) = \mathsf{H}(g^{s'} \cdot \mathsf{pk}_x^{-\prime c'\cdot y'} \| g^{t'} \cdot \mathsf{pk}_z'^{y'} \| m)$ by showing that

$$g^{s^*} \cdot \mathsf{pk}_x^{c^*\cdot y^*} = g^{s'} \cdot \mathsf{pk}_x'^{-c'\cdot y'}$$

and

$$g^{t^*} \cdot \mathsf{pk}_z^{y^*} = g^{t'} \cdot \mathsf{pk}_z'^{y'}.$$

We prove the first statement as follows

$$\begin{aligned} g^{s^*} \cdot \mathsf{pk}_x^{c^*\cdot y^*} =& g^{s'-\mathsf{ck}_x\cdot c\cdot y} \cdot \mathsf{pk}_x^{c'\cdot y'} && (60)\\ =& g^{s'} \cdot (\mathsf{pk}_x \cdot g^{\mathsf{ck}_x})^{-c'\cdot y'} && (61)\\ =& g^{s'} \cdot \mathsf{pk}_x'^{-c'\cdot y'} && (62)\end{aligned}$$

and the second statement as follows:

$$\begin{aligned} g^{t^*} \cdot \mathsf{pk}_z^{y^*} =& g^{t'+\mathsf{ck}_z\cdot y} \cdot \mathsf{pk}_z^{y'} && (63)\\ =& g^{t'} \cdot (\mathsf{pk}_z \cdot g^{\mathsf{ck}_z})^{y'} && (64)\\ =& g^{t'} \cdot \mathsf{pk}_z'^{y'} && (65)\end{aligned}$$

□

References

1. Abe, M. *A Secure Three-Move Blind Signature Scheme for Polynomially Many Signatures* in *EUROCRYPT 2001* (2001).
2. Abe, M. & Okamoto, T. *Provably Secure Partially Blind Signatures* in *CRYPTO 2000* (2000).
3. Alkeilani Alkadri, N., El Bansarkhani, R. & Buchmann, J. *BLAZE: Practical Lattice-Based Blind Signatures for Privacy-Preserving Applications* in *FC 2020* (2020).
4. Bellare, M., Boldyreva, A. & Micali, S. *Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements* in *EUROCRYPT 2000* (2000).
5. Bellare, M. & Rogaway, P. *The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs* in *EUROCRYPT 2006* (2006).
6. Blake-Wilson, S. & Menezes, A. *Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol* in *PKC'99* (1999).
7. Boldyreva, A. *Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme* in *PKC 2003* (2003).
8. Boneh, D., Lynn, B. & Shacham, H. *Short Signatures from the Weil Pairing* in *ASIACRYPT 2001* (2001).
9. Brands, S. *Restrictive Blinding of Secret-Key Certificates* in *EUROCRYPT'95* (1995).
10. Chaum, D. *Blind Signatures for Untraceable Payments* in *CRYPTO'82* (1982).
11. Chaum, D. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Commun. ACM* (1985).
12. Fischlin, M. *Round-Optimal Composable Blind Signatures in the Common Reference String Model* in *CRYPTO 2006* (2006).
13. Fischlin, M. & Schröder, D. *Security of Blind Signatures under Aborts* in *PKC 2009* (2009).
14. Galbraith, S. D., Malone-Lee, J. & Smart, N. P. Public key signatures in the multi-user setting. *Inf. Process. Lett.* (2002).
15. Gjøsteen, K. & Kråkmo, L. *Round-Optimal Blind Signatures from Waters Signatures* in *ProvSec 2008* (2008).
16. Hanaoka, G. & Schuldt, J. C. N. *On signatures with tight security in the multi-user setting* in *2016 International Symposium on Information Theory and Its Applications, ISITA 2016, Monterey, CA, USA, October 30 - November 2, 2016* (2016).
17. Hauck, E., Kiltz, E., Loss, J. & Nguyen, N. K. *Lattice-Based Blind Signatures, Revisited* in *CRYPTO 2020, Part II* (2020).
18. Hofheinz, D. & Nguyen, N. K. *On Tightly Secure Primitives in the Multi-instance Setting* in *PKC 2019, Part I* (2019).
19. Menezes, A. & Smart, N. P. Security of Signature Schemes in a Multi-User Setting. *Des. Codes Cryptogr.* (2004).
20. Okamoto, T. *Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes* in *CRYPTO'92* (1993).
21. Pointcheval, D. & Stern, J. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology* (2000).
22. Rivest, R. L., Shamir, A. & Adleman, L. M. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* (1978).
23. Rückert, M. *Lattice-Based Blind Signatures* in *ASIACRYPT 2010* (2010).

24. Schnorr, C.-P. *Security of Blind Discrete Log Signatures against Interactive Attacks* in *ICICS 01* (2001).
25. Tessaro, S. & Zhu, C. *Short Pairing-Free Blind Signatures with Exponential Security* in *EUROCRYPT 2022, Part II* (2022).

Supplementary Material

A Additional Preliminaries

Definition 16 (Partial Blindness). *For a partially blind signature scheme Σ and an adversary $\mathcal{A}$ we define the blindness game as follows:*

```
$\mathsf{PartiallyBlind}_{\Sigma}^{\mathcal{A}}$
1 : $(\mathsf{pk}_{BS}, m_0, m_1, \mathsf{info}, \mathsf{state}_{find}) \xleftarrow{\$} \mathcal{A}(find)$
2 : $b \xleftarrow{\$} \{0,1\}$
3 : $\mathsf{state}_{issue} \xleftarrow{\$} \mathcal{A}^{\langle \cdot, \mathsf{User}(\mathsf{pk}_{BS}, m_b, \mathsf{info})\rangle^1, \langle \cdot, \mathsf{User}(\mathsf{pk}_{BS}, m_{1-b}, \mathsf{info})\rangle^1}(issue, \mathsf{state}_{find})$
4 :     let $\sigma_b$ the final output of $\mathsf{User}(\mathsf{pk}_{BS}, m_b, \mathsf{info})$,
5 :     $\sigma_{1-b}$ the final output of $\mathsf{User}(\mathsf{pk}_{BS}, m_{1-b}, \mathsf{info})$.
6 : if $(\sigma_0 = \perp \vee \sigma_1 = \perp)$
7 :     $(\sigma_0, \sigma_1) = (\perp, \perp)$
8 : $b^* \leftarrow \mathcal{A}(guess, (m_0, \sigma_0, \mathsf{info}), (m_1, \sigma_1, \mathsf{info}))$
9 : return $b = b^*$.
```

We define the advantage

$$\mathsf{Adv}_{\Sigma,\mathcal{A}}^{\text{PartiallyBlind}} := 2 \cdot \left| \frac{1}{2} - \Pr\left[\mathsf{PartiallyBlind}_{\Sigma}^{\mathcal{A}} = 1\right] \right|$$

and we say that the scheme Σ is (t, ε)-partially-blind if for all adversaries $\mathcal{A}$ that run in time at most t it holds that $\mathsf{Adv}_{\Sigma,\mathcal{A}}^{\text{PartiallyBlind}} \leq \varepsilon$.

B Figures of Signing Interactions of Blind Signature Schemes

Signer		**User**
$\mathsf{sk} = x$		$\mathsf{pk} = y$
		m
		$r \xleftarrow{\$} \mathbb{Z}_p$
	$\xleftarrow{\overline{m}}$	$\overline{m} \leftarrow \mathsf{H}(m) \cdot g^r$
$\overline{\sigma} \leftarrow (\overline{m})^x$	$\xrightarrow{\overline{\sigma}}$	
		$\sigma \leftarrow \overline{\sigma} \cdot y^{-r}$
		$\Downarrow$
		(m, σ)

Fig. 13. The Blind BLS Signature Scheme

Signer		**User**
$\mathsf{sk} = x$		$\mathsf{pk} = y = g^x$
		m
$r \xleftarrow{\$} \mathbb{Z}_p$		
$R \leftarrow g^r$	$\xrightarrow{R}$	
		$\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$
		$R' \leftarrow R \cdot g^\alpha \cdot y^\beta$
		$c' \leftarrow \mathsf{H}(R', m)$
	$\xleftarrow{c}$	$c \leftarrow c' + \beta$
$s \leftarrow c \cdot x + r$	$\xrightarrow{s}$	
		$g^s \stackrel{?}{=} R \cdot y^c$
		$s' \leftarrow s + \alpha$
		$\Downarrow$
		$(m, (R', s'))$

Fig. 14. Schnorr's Blind Signature Scheme

Signer		**User**
$\mathsf{sk} = (r, s)$		$\mathsf{pk} = y = g^{-r} \cdot h^{-s}$
		m
$t, u \xleftarrow{\$} \mathbb{Z}_p$		
$a \leftarrow g^t \cdot h^u$	$\xrightarrow{\quad a \quad}$	
		$\beta, \gamma, \delta \xleftarrow{\$} \mathbb{Z}_p$
		$\alpha \leftarrow a \cdot g^\beta \cdot h^\gamma \cdot y^\delta$
		$\varepsilon \leftarrow \mathsf{H}(m, \alpha)$
	$\xleftarrow{\quad e \quad}$	$e \leftarrow \varepsilon - \delta$
$R \leftarrow t + e \cdot r$		
$S \leftarrow u + e \cdot s$	$\xrightarrow{\quad R, S \quad}$	
		$a \overset{?}{=} g^R \cdot h^S \cdot y^e$
		$\rho \leftarrow R + \beta$
		$\sigma \leftarrow S + \gamma$
		$\Downarrow$
		$(m, (\alpha, \rho, \sigma))$

Fig. 15. The Okamoto-Schnorr Blind Signature Scheme

Signer		**User**
$\mathsf{sk} = x$		$\mathsf{pk} = X = g^x$
		m
$a \xleftarrow{\$} \mathbb{Z}_p$		$r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$
$y \xleftarrow{\$} \mathbb{Z}_p^*$		$\gamma \xleftarrow{\$} \mathbb{Z}_p^*$
$A \leftarrow g^a$	$\xrightarrow{(A, \mathrm{Y})}$	$\mathrm{Y}' \leftarrow \mathrm{Y}^\gamma$
$\mathrm{Y} \leftarrow X^y$		$A' \leftarrow g^{r_1} \cdot A^\gamma \cdot \mathrm{Y}'^{r_2}$
		$c' \leftarrow \mathsf{H}(A' \Vert \mathrm{Y}' \Vert m)$
	$\xleftarrow{c}$	$c \leftarrow c' + r_2$
$s \leftarrow a + c \cdot y \cdot x$	$\xrightarrow{(s, y)}$	$y \overset{?}{\neq} 0, \mathrm{Y} \overset{?}{=} X^y$
		$g^s \overset{?}{=} A \cdot \mathrm{Y}^c$
		$s' \leftarrow \gamma \cdot s + r_1$
		$y' \leftarrow \gamma \cdot y$
		$\Downarrow$
		$(m, (c', s', y'))$

Fig. 16. Tessaro-Zhu GGM Blind Signature scheme where $\mathsf{pp}=$ (p,g,H)

Signer		**User**
$\mathsf{sk} = x$		$\mathsf{pk} = (X = g^x, Z)$
		m
$a, t \xleftarrow{\$} \mathbb{Z}_p$		$r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p$
$y \xleftarrow{\$} \mathbb{Z}_p^*$		$\gamma_1, \gamma_2 \xleftarrow{\$} \mathbb{Z}_p^*$
$A \leftarrow g^a$	$\xrightarrow{(A, C)}$	$C' \leftarrow C^{\gamma_1} \cdot g^{r_2}$
$C \leftarrow g^t \cdot Z^y$		$A' \leftarrow g^{r_1} \cdot A^{\gamma_1/\gamma_2}$
		$c' \leftarrow \mathsf{H}(A' \Vert C' \Vert m)$
	$\xleftarrow{c}$	$c \leftarrow c' \cdot \gamma_2$
$c \overset{?}{\neq} 0$		
$s \leftarrow a + c \cdot y \cdot x$	$\xrightarrow{(s, y, t)}$	$y \overset{?}{\neq} 0, C \overset{?}{=} g^t \cdot Z^y$
		$g^s \overset{?}{=} A \cdot X^{c \cdot y}$
		$s' \leftarrow (\gamma_1/\gamma_2) \cdot s + r_1$
		$y' \leftarrow \gamma_1 \cdot y$
		$t' \leftarrow \gamma_1 \cdot t + r_2$
		$\Downarrow$
		$(m, (c', s', y', t'))$

Fig. 17. Tessaro-Zhu AGM Blind Signature scheme where pp= (p, $\mathbb{G}$,g,H)