

SoK: Signatures With Randomizable Keys^{*}

Sofia Celi¹, Scott Griffy², Lucjan Hanzlik³, Octavio Perez-Kempner⁴, and Daniel Slamanig⁵

¹ Brave Software, sceli@brave.com

² Brown University, Providence, USA, scott_griffy@brown.edu

³ CISP Helmoltz Center for Information Security, Saarbrücken, Germany, hanzlik@cispa.de

⁴ NTT Social Informatics Laboratories, Tokyo, Japan, octavio.perezkempner@ntt.com

⁵ Research Institute CODE, Universität der Bundeswehr München, München, Germany, daniel.slamanig@unibw.de

Abstract. Digital signature schemes with specific properties have recently seen various real-world applications with a strong emphasis on privacy-enhancing technologies. They have been extensively used to develop anonymous credentials schemes and to achieve an even more comprehensive range of functionalities in the decentralized web.

Substantial work has been done to formalize different types of signatures where an allowable set of transformations can be applied to message-signature pairs to obtain new related pairs. Most of the previous work focused on transformations with respect to the message being signed, but little has been done to study what happens when transformations apply to the signing keys. A first attempt to thoroughly formalize such aspects was carried by Derler and Slamanig (ePrint’16, Designs, Codes and Cryptography’19), followed by the more recent efforts by Backes *et al.* (ASIACRYPT’18) and Eaton *et al.* (ePrint’23). However, the literature on the topic is vast and different terminology is used across contributions, which makes it difficult to compare related works and understand the range of applications covered by a given construction.

In this work, we present a unified view of *signatures with randomizable keys* and revisit their security properties. We focus on state-of-the-art constructions and related applications, identifying existing challenges. Our systematization allows us to highlight gaps, open questions and directions for future research on signatures with randomizable keys.

Keywords: Digital signatures, key blinding, key randomization

1 Introduction

Digital signatures are an invaluable cryptographic primitive for the authenticity and integrity of data. Over the years, different variants with advanced properties have been introduced. One particular primitive used for privacy-enhancing applications [CGT23] (e.g., anonymity networks, rate-limiting applications, deterministic wallets and stealth addresses) are *malleable* signatures. These are schemes where given a signature σ on a message m , one can efficiently derive a new signature σ' on a message m' . Following the notation by Chase *et al.* [CKLM14], a digital signature is *malleable* if, on input a message m and a signature σ , it is possible to efficiently compute a signature σ' on a related message $m' = T(m)$, for a (n -ary) transformation T allowed by the signature scheme. Ideally, σ' should look like a freshly computed signature on m' , a property known as *context hiding*.

The study of malleable signatures originates from the work on *homomorphic signatures* [JMSW02], and subsequent formalizations by Ahn *et al.* [ABC⁺12] and Attrapadung *et al.* [ALP12]. They can be considered a generalization of existing primitives, such as quotable [ABC⁺12, HHH⁺08] or redactable signatures [SBZ02, JMSW02, BBD⁺10], homomorphic signatures for restricted classes [BFKW09, BF11, CFW14] or any classes of functions [GVW15] (cf. [DDH⁺15] for a comprehensive overview). When T is unary, i.e., it operates on a single signature, one obtains primitives such as quotable or redactable signatures. For n -ary transformations, i.e., T operates on $n > 1$ signatures, one obtains homomorphic signatures (cf. [CKLM14] for a more detailed discussion). In this work, we only consider the unary case. One particular class of such signatures is *randomizable signatures* [CL03, CL04, PS16]. They allow one to maul the signature but leave the message untouched, i.e., $m' = m$, and one can publicly derive a new signature that is distributed like a fresh one on m' . Thus, they can be seen as a special case where T is the identity function, and the scheme provides context hiding.

^{*} This is the full version of [CGH⁺24]. In case of citing our work, please cite the proceedings version.

To the best of our knowledge, the first work to explicitly study malleability on the key space is the work by Derler and Slamanig [DS16, DS19] on *key-homomorphic* signatures inspired by previous works on pseudo-random functions [BLMR13] and encryption [BGG⁺14]. While they discuss the unary and the n -ary case, again, we only focus on the former. They consider secret and public key spaces to be groups (with an efficiently computable homomorphism μ from the secret to the public key space), and the functionality that a given signature σ for message m that verifies under pk can be adapted to a signature σ' on m under pk' . The functionality is obtained by applying μ to a randomly sampled element from the secret key space and combining the result with pk .

The concept of signatures with *re-randomizable keys* was introduced earlier in the context of sanitizable signatures by Fleischhacker *et al.* in [FKM⁺16] and subsequently used in [DFL19, ADE⁺20] and [ER22]. Kiltz *et al.* [KMP16] have used *key-rerandomizability* (calling it “random self-reducibility”) of canonical identification schemes when converted to signature schemes. Another similar notion that appears in the literature is known as *key-blinding* (sometimes referred to as *key-randomization*). This notion has recently been used and further formalized in the works by Eaton *et al.* [ESS21] and [ELW23]. Backes *et al.* [BHKs18] introduce signature schemes with flexible public keys, also focusing on re-randomizable keys. But instead of switching keys arbitrarily, secret and public keys live in equivalence classes induced by a relation $\mathcal{R}$, and keys are re-randomized between representatives of the respective classes. This relation is usually chosen based on an underlying computationally hard problem to ensure any form of unlinkability. We also note that malleability of signature schemes w.r.t. the message and key spaces have been studied under the name of *mercurial signatures* in [CL19] and [CLPK22]. Similar notions have also been studied in works targeting related-key attacks (*e.g.*, [BCM11, BPT12, MSM⁺16]).

Ferreira and Dahab [FD02, FD04] construct schemes named *blinded-key* signatures as a means to protect a long-term secret key from being stolen. While their goal looks similar to that of *one-way* blinding from [ESS21], it considers a different case where the adversary receives a blinded secret key and must recover the long-term secret key. Furthermore, [FD02] and [FD04] do not provide formal definitions and rely on a trusted third party for signature verification, making it difficult to compare with the previous works (albeit the naming convention is similar).

There is also recent work on key-updatable signatures [JS18] and key-updating signatures [JMM19]. These signatures support updating of secret keys and corresponding public keys, and are generalizations of (or closely related to) forward-secure (or key-evolving) signatures [BM99]. Their main focus is on unforgeability under certain key leakages and are used in the construction of strongly secure messaging protocols. However, they are not interested in updated keys being indistinguishable from freshly generated keys, nor in the randomization or adaption of issued signatures. More recently, the notion of updatable and randomizable signatures has been introduced in context of the same application in [DHRR22]. Such schemes support asynchronous updates of secret and public keys as well as randomization of secret keys, where for the latter indistinguishability is required. Nevertheless, as above, there is no notion of adaption, making it incomparable.

Our approach and contributions. We aim to present a complete overview of *signatures with randomizable keys*, which we see as malleable signatures w.r.t. the key space. To systematize their knowledge, we review existing works on the topic and revisit security definitions with a focus on privacy-preserving applications. As a result, we propose new security notions to better capture different attack scenarios and adversarial behaviour. As applications evolve, we aim for a general security framework to capture all the possible combinations. More specifically, we propose a set of parametrized definitions to capture *unforgeability*, *unlinkability* and *unextractability* of signature schemes. The latter notion differentiates from *unlinkability* in that the adversary is challenged to *extract* the long-term public key when given access to randomizations of it and corresponding key-randomizers. This contrasts with *unlinkability* where access to the long-term public key and randomizations of it is given but not the key-randomizer. Furthermore, our formalizations also consider maliciously generated parameters, keys and oracles in a comprehensive manner. From there, we identify research gaps and discuss related challenges as part of our contributions. Along the way, we also show how definitions given in previous works can be strengthened.

2 Preliminaries

To introduce the different terminology and related concepts of signatures with randomizable keys, we follow the approach of [DS19]. However, unlike [DS19], we do not consider combinations of keys nor take into account the particular structure of a given homomorphism. Instead, we opt to abstract the idea of “allowable” transformations to provide more general definitions. This is the main reason to propose a different name and a slightly different formalization. In the following, we present the required notation and basic definitions of signatures with randomizable keys (SWRK).

Notation. PPT stands for probabilistic polynomial time. We use λ to denote the security parameter; $\epsilon(\lambda)$ for a negligible function. $r \leftarrow \$ \mathcal{S}$ denotes sampling r from set $\mathcal{S}$ uniformly at random. We write $\mathcal{A}(x; y)$ to specify that $\mathcal{A}$ uses randomness y on input x . Similarly, $\mathcal{A}(x, [y])$ indicates that y is an optional parameter. We denote the signature (resp. message, public-key, secret-key and key-randomizer) space by SIG (resp. $\mathcal{M}$, $\mathcal{PK}$, $\mathcal{SK}$ and $\mathcal{KR}$).

Definition 1 (Signatures With Randomizable Keys). A signature scheme with randomizable keys (SWRK) consists of the following algorithms:

$\text{PPGen}(1^\lambda)$ is a PPT algorithm that, given λ , outputs public parameters pp .

$\text{KGen}(\text{pp})$ is a PPT algorithm that, given pp , outputs a key pair (sk, pk) .

$\text{Sign}(\text{pp}, \text{sk}, m)$ is a PPT algorithm that, given pp , a message m and a secret key sk , outputs a signature σ on m .

$\text{Verify}(\text{pp}, m, \sigma, \text{pk})$ is a deterministic algorithm that takes as input pp , m , σ , and public key pk . It outputs 1 if and only if σ is a valid signature on m .

$\text{RandPK}(\text{pp}, \text{pk}, \rho)$ is a PPT algorithm that, given pp , a key randomizer ρ , and pk , outputs a new public key pk' s.t. $\text{pk}' = T(\text{pk}, \rho)$ for some transformation T .

$\text{RandSK}(\text{pp}, \text{sk}, \rho)$ is a PPT algorithm that, given pp , ρ , and sk , outputs a new secret key sk' s.t. $\text{sk}' = \tilde{T}(\text{sk}, \rho)$ for some transformation $\tilde{T}$.

$\text{Adapt}(\text{pp}, [m], \sigma, \rho, \text{pk})$ is a PPT algorithm that takes as input pp , m (optional), σ , ρ , and pk . It computes an adapted signature σ' under a new public key pk' s.t. $\text{pk}' = T(\text{pk}, \rho)$ for some transformation T and outputs (pk', σ') .

$\text{VerKey}(\text{pp}, \text{sk}, \text{pk})$ is a deterministic algorithm that takes as input pp and a key pair (sk, pk) . If (sk, pk) is a valid key pair it outputs 1 and 0 otherwise.

Security requires the scheme to be at least correct and unforgeable. To support the **Adapt** algorithm, SWRK should provide adaptability in the sense of the aforementioned context hiding notion and (perfect) adaption correctness (i.e., signatures adapted with **Adapt** should verify as long as the original signature does). However, there are SWRK schemes that support key randomization but not adaption. While one could distinguish between SWRK and aSWRK (adaptable SWRK), for simplicity, we keep the term SWRK and assume the schemes provide adaption unless otherwise explicitly mentioned. In what follows, we present these properties based on the literature. In Sec. 4, we revisit them and include new ones as part of our systematization effort.

Definition 2 (Correctness). A SWRK scheme is correct if for every security parameter λ , message m s.t. $\text{pp} \leftarrow \$ \text{PPGen}(1^\lambda)$ and $(\text{sk}, \text{pk}) \leftarrow \$ \text{KGen}(\text{pp})$: $\Pr[\sigma \leftarrow \text{Sign}(\text{sk}, m) : \text{Verify}(m, \sigma, \text{pk}) = 1] = 1$.

Definition 3 (EUF-CMA). A SWRK scheme is existentially unforgeable under adaptively chosen-message attacks, if for all PPT adversaries $\mathcal{A}$ with access to a signing oracle Sign , the following probability is negligible,

$$\Pr \left[\begin{array}{l} \text{pp} \leftarrow \$ \text{PPGen}(1^\lambda); (\text{sk}, \text{pk}) \leftarrow \$ \text{KGen}(\text{pp}), \\ (m^*, \sigma^*) \leftarrow \$ \mathcal{A}^{\text{Sign}(\text{sk}, \cdot)}(\text{pk}) \end{array} : \begin{array}{l} \forall m \in Q : m^* \neq m \wedge \\ \text{Verify}(m^*, \sigma^*, \text{pk}) = 1 \end{array} \right],$$

where Q is the set of queries that $\mathcal{A}$ has issued to the signing oracle.

For unforgeability, one can also consider the strong variant in which the oracle keeps track of message-signature pairs and the adversary wins if $(m^*, \sigma^*) \notin Q$.

Definition 4 (Signature Adaption). A SWRK scheme provides signature adaption if, for every security parameter λ , message m , and key randomizer ρ , the advantage of any adversary $\mathcal{A}$ defined by $\text{Adv}_{\Gamma, \mathcal{A}}^{\text{Adapt}}(\lambda) := 2 \cdot \Pr[\text{Exp}_{\Gamma, \mathcal{A}}^{\text{Adapt}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\text{Exp}_{\Gamma, \mathcal{A}}^{\text{Adapt}}(\lambda)$ is shown in Fig. 1.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{Adapt}}(\lambda)$

$\text{pp} \leftarrow \$ \text{PPGen}(1^\lambda); b \leftarrow \$ \{0, 1\}; (\text{sk}, \text{pk}) \leftarrow \$ \text{KGen}(\text{pp}); \rho_0, \rho_1 \leftarrow \$ \mathcal{KR}; \sigma \leftarrow \text{Sign}(\text{sk}, m)$
 $\tau_0 \leftarrow \text{Adapt}(m, \sigma, \rho_0, \text{pk}); \tau_1 \leftarrow (\text{RandPK}(\text{pk}, \rho_1), \text{Sign}(\text{RandSK}(\text{sk}, \rho_1), m))$
 $b' \leftarrow \$ \mathcal{A}(\tau_b); \text{return } b = b'$

Fig. 1. Signature adaption experiment.

A stronger notion for signature adaption is called *perfect adaption* (see Definition 5) and it states that distinguishing between a fresh signature and an adapted signature should be hard even when the original signature is known to the adversary. Both notions can be stated with unconditional security against unbounded adversaries or restricted to computational security (PPT adversaries).

Definition 5 (Perfect Adaption). *A SWRK scheme provides perfect adaption if, for every security parameter λ , message m , and key randomizer ρ , it holds that $\{\sigma, \text{Adapt}(m, \sigma, \rho, \text{pk})\}$ and $\{\sigma, \text{RandPK}(\text{pk}, \rho), \text{Sign}(\text{RandSK}(\text{sk}, \rho), m)\}$ are identical distributions where $\text{pp} \leftarrow \$ \text{PPGen}(1^\lambda)$, $b \leftarrow \$ \{0, 1\}$, $(\text{sk}, \text{pk}) \leftarrow \$ \text{KGen}(\text{pp})$, and $\sigma \leftarrow \text{Sign}(\text{sk}, m)$.*

The previous definitions for signature adaption consider *honest parameters* and *honestly generated keys*. However, one can also address what happens when any (or both) of the previous is maliciously generated as in [FHS19, KSD19, CLPK22].

3 Literature Review

We present an overview of previous work on the topic, unifying syntax and notation in accordance to Sec. 2. We also compare and discuss the shortcomings of each work whenever it corresponds, setting the grounds for our systematization.

3.1 Equivalence Class Signatures

While the prime focus of Structure-Preserving Signatures on Equivalence Classes (*i.e.*, SPS-EQ or Equivalence Class Signatures) [HS14, FHS19] is message randomization, they have inspired and have been used to build SWRK. We identify two lines of work in this regard. The first one is *mercurial signatures* [CL19], which are malleable signatures that allow transformations on all: the message, signature and key spaces. Consequently, they can be viewed as SWRK whenever the message is not randomized. There are two constructions: i) [CL19] (based on [FHS19]) in the generic group model (GGM), and ii) [CLPK22] (based on [KSD19]) in the standard model. The second line of work is *signatures with flexible public keys* (SFPK [BHKS18]), which solely focuses on equivalence classes on the key space and, thus, key randomization. In both mercurial signatures and SFPK, any individual message or key is actually a class *representative*, and privacy and unforgeability are defined over these classes. SFPK constructions exist under standard assumptions [BHKS18], in the common reference string model (CRS, [BHKS18]) and in the random oracle model (ROM, [HS21]). Regardless of the specific equivalence relation in which an SPS-EQ acts, they all require some form of signature adaption (as in Def. 4), which should also provide unlinkability with respect to a class. This stronger property has been studied with some variations and referred to under different names such as *class-hiding*, *origin-hiding*, as well as *perfect adaption* when the context is clear. In the following, we present the relevant work on signatures acting on equivalence classes of keys alongside the formalizations of such variations.

Adaption in Mercurial Signatures. The original definition of *class-hiding* for SPS-EQ from [HS14] focused on messages and signatures. It only considered that given an *honestly generated signature*, changing the message using randomness μ and adapting the signature should look like a random message-signature pair. The subsequently developed notion of *perfect adaption* [FHS15, FHS19, KSD19] explicitly states that adapting a message-signature pair with randomness μ should look like a fresh signature for the same message. This notion was also extended to consider potentially maliciously generated parameters, signatures, and keys.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-MS}}(\lambda)$	Oracle $\text{Sign}(\text{sk}, m)$
$\text{pp} \leftarrow \text{PPGen}(1^\lambda); b \leftarrow \{0, 1\}; \rho \leftarrow \mathcal{KR}$	return $\text{Sign}(\text{sk}, m)$
$(\text{sk}_1, \text{pk}_1) \leftarrow \text{KGen}(\text{pp}); (\text{sk}_2^0, \text{pk}_2^0) \leftarrow \text{KGen}(\text{pp})$	
$\text{pk}_2^1 \leftarrow \text{RandPK}(\text{pk}_1, \rho); \text{sk}_2^1 \leftarrow \text{RandSK}(\text{sk}_1, \rho)$	
$b' \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}_1, \cdot), \text{Sign}(\text{sk}_2^b, \cdot)}(\text{pk}_1, \text{pk}_2^b); \text{return } b = b'$	

Fig. 2. Public key class-hiding experiment from [CL19].

Mercurial signatures include three functions to randomize keys and signatures, `RandPK`, `RandSK`, and `ConvertSig`. They also provide a fourth function, `ChangeRep` which randomizes the message space and signature together, but our systematization does not consider message space randomizations.

The `ConvertSig` algorithm in [CL19] outputs a valid signature which will verify under a new representative of a public key class, randomizing the signature but leaving the message representative unchanged. The public key can then be randomized with `RandPK` to output a new public key which correctly verifies with the new signature from `ConvertSig` when the same key randomizer (called a “key converter” in their work) is supplied to both functions (`ConvertSig` and `RandPK`). We can consider our `Adapt` function from Sec. 2 as running the two functions simultaneously. In [CL21], the `ConvertSig` algorithm is extended to also randomize the message representative alongside the signature. If the identity of the message space is passed to `ConvertSig` in [CL21], it will operate exactly like the `ConvertSig` from [CL19]. Adaptability for mercurial signatures is formalized as *origin-hiding for ConvertSig* (Def. 6) and *origin-hiding for ChangeRep* in [CL19].

Definition 6 (Origin-Hiding for ConvertSig [CL19]). A mercurial signature scheme, Γ , is *origin-hiding for ConvertSig* if, given any tuple (pk, σ, m) that verifies, and given a random key randomizer ρ , $\text{ConvertSig}(\sigma, \text{pk}, \rho)$ outputs a new signature σ' such that σ' is a uniform random signature in SIG for m , i.e., $\sigma' \in \{\sigma^* \mid \text{Verify}(\text{RandPK}(\text{pk}, \rho), m, \sigma^*) = 1\}$.

We also observe that the notion of origin-hiding was formalized in a slightly different way in [CLPK22] where the authors follow the terminology from [FHS19, KSD19]. In that work, a definition is given for perfect adaption of signatures w.r.t. the key space under maliciously generated keys in the honest parameters model. As in [CL19], the construction from [CLPK22] only achieves a weaker notion of perfect adaption under *honestly* generated keys in the honest parameter model. To the best of our knowledge, it remains an open problem to build a mercurial signature scheme with perfect adaption under maliciously generated keys.

Inspired by previous work, [CL19] considered the notion of *public key class-hiding* (Def. 7). It states that an adversary cannot succeed in determining whether message-signature pairs are being generated from keys in the same equivalence class, i.e., valid key randomizations, or distinct equivalence classes.

Definition 7 (Public Key Class-Hiding [CL19]). A mercurial signature scheme has *public key class-hiding* if the advantage of any PPT adversary $\mathcal{A}$ defined by $\text{Adv}_{\Gamma, \mathcal{A}}^{\text{PKCH-MS}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-MS}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-MS}}(\lambda)$ is shown in Fig. 2.

Definition 7 is similar to *class-hiding* from [FHS19] as both challenge the adversary to distinguish whether two values are from the same equivalence class or not. However, in Def. 7, the values are keys (instead of messages), so the adversary is additionally given access to signing oracles. Both definitions assume that the values (keys and messages) are honestly generated.

As we show in Sec. 4, our unlinkability notion (Def. 16) captures the idea behind public key class-hiding. Moreover, we extend it by considering all possible signing oracles and maliciously generated parameters and keys.

Signatures with flexible public keys. *Signatures with flexible public keys* (SFPK) introduced by Backes *et al.* [BHKS18] consider equivalence classes solely on the public key space. One of the goals of the authors was to construct a SFPK scheme for which the public key space would be the message space of existing equivalence class signatures, i.e., a vector of group elements of one of the pairing groups. The authors combined their SFPK scheme with the equivalence class signature scheme from [HS14], which allowed them to build short static group signatures, and the first sublinear ring signature scheme without ROM or a CRS.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-SFPK}}(\lambda)$	Oracle $\text{Sign}(\text{sk}, m)$
$\text{pp} \leftarrow \text{PPGen}(1^\lambda); b \leftarrow \{0, 1\}; \rho \leftarrow \mathcal{KR}$	return $\text{Sign}(\text{sk}, m)$
$\{\text{sk}_i, \text{pk}_i\}_{i \in \{0, 1\}} \leftarrow \text{KGen}(\text{pp})$	
$\text{pk}' \leftarrow \text{RandPK}(\text{pk}_b, \rho); \text{sk}' \leftarrow \text{RandSK}(\text{sk}_b, \rho)$	
$b' \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}', \cdot)}(\{\text{sk}_i, \text{pk}_i\}_{i \in \{0, 1\}}, \text{pk}'); \text{return } b = b'$	

Fig. 3. Class-hiding experiment from [BHKS18].

The *class-hiding* notion introduced in [BHKS18] gives the adversary access to the random coins used by KGen. This strong notion makes the primitive useful in constructing ring signatures satisfying a strong anonymity property (*i.e.*, anonymity against full key exposure [BKM09]). This idea contrasts with the properties of mercurial signatures, as the latter do not fulfill this strong notion of class-hiding (*i.e.*, an adversary can recognize a public key using the secret key). However, the relaxed property has been proven useful in some settings, *e.g.*, [GL24], allowing signers to recognize their randomized public key from a key/signature pair but without being able to link the signature to a particular signing process. We present the basic definition from [BHKS18] (*i.e.*, without access to the random coins).

Definition 8 (Class-Hiding [BHKS18]). A SFPK scheme, Γ , has class-hiding if the advantage of any PPT adversary $\mathcal{A}$ in the $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-SFPK}}$ experiment is negligible as defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{PKCH-SFPK}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-SFPK}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{PKCH-SFPK}}(\lambda)$ is shown in Fig. 3.

SFPK can also be built in a way that class-hiding is conditional. In this line, Backes *et al.* [BHKS18] define an alternative key generation algorithm, TKGen, that outputs an additional trapdoor alongside the key pair. The former allows the identification of any public key in relation to the key pair but class-hiding holds as long as this trapdoor is unknown. This trapdoor is also used by the challenger in the unforgeability experiment to verify the winning conditions, *i.e.*, that the public key output by the adversary is in relation to the challenged key. Keypairs generated by KGen and TKGen should be indistinguishable for all the previous conditions to hold. This indistinguishability allowed Backes *et al.* to define class-hiding w.r.t. the former and unforgeability regarding the latter. Unforgeability holds even if the adversary is given the trapdoor. But, linking public keys using the corresponding secret key cannot be secure in such a case.

Backes *et al.* [BHKS18] also introduce the notion of *key recovery*, allowing the signer to retrieve the signing key for public keys generated by third parties, *i.e.*, via randomizing the original key. They show that this property has applications for stealth addresses and give a construction in the standard model. Unfortunately, due to its specific public key structure, the construction cannot be combined with the SPS-EQ of [HS14]. In the same paper, the authors define SFPK in the presence of multiple signers, modeled as a setup algorithm for generating public parameters for all signers. For this case, they consider the properties of class-hiding and unforgeability in the presence of maliciously generated parameters.

In follow-up work, Backes *et al.* [BHSB19] introduce a weaker notion called *class-hiding with key corruption*. Instead of giving the random coins used in KGen, the challenger provides just the secret keys for the challenged public keys. This weaker notion allowed for a more efficient SFPK scheme (a construction that is enough for their group signature application). The authors also proposed the idea of a *canonical representative*, *i.e.*, a distinct public key for each relation that represents the whole class. An example representative used in [BHSB19] consists of a vector of group elements, where the first element is a specified group generator.

Lastly, Hanzlik and Slamanig [HS21] used the same combination of SFPK and SPS-EQ (as in [BHKS18] and [BHSB19]) to construct efficient anonymous credentials. The authors also introduce the idea of split signing for SFPK, a technique allowing the distribution of the signing process between two parties. In brief, one party performs the essential operations (*i.e.*, using the secret key) while the other performs computationally inefficient operations without requiring the secret key. This technique allowed Hanzlik and Slamanig to run parts of the SFPK signing process on a constrained smart card while the user's smartphone performs the more computationally complex operation without being able to sign by itself.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNF}}(\lambda)$

$\Sigma \leftarrow \emptyset; \text{pp} \leftarrow \text{PPGen}(1^\lambda); (\text{sk}, \text{pk}) \leftarrow \text{KGen}(\text{pp}); (\rho^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}, \cdot, \cdot)}(\text{pk})$
 $\text{pk}^* \leftarrow \text{RandPK}(\text{pk}, \rho^*); \text{return } m \notin \Sigma \wedge (\text{Verify}(m^*, \sigma^*, \text{pk}) \vee \text{Verify}(m^*, \sigma^*, \text{pk}^*))$

Oracle $\text{Sign}(\text{sk}, m, \rho)$

$\Sigma \leftarrow \Sigma \cup \{m\}; \text{if } \rho = \perp \text{ return } \text{Sign}(\text{sk}, m); \text{return } \text{Sign}(\text{RandSK}(\text{sk}, \rho), m)$

Fig. 4. Unforgeability experiment from [FKM⁺16].

3.2 Signatures with Re-Randomizable Keys

Fleischhacker *et al.* [FKM⁺16] introduce signatures with (*perfect*) *re-randomizable keys*. Such a signature scheme allows to “re-randomize” (or simply randomize) both the signing and the verification key separately; but, it is required that the re-randomization is perfect (re-randomized keys must have the same distribution as the original ones). Their main motivation is to construct *sanitizable signatures* which allow a signer to authenticate a message so that another dedicated party (the sanitizer) can modify parts of it without invalidating the signature.

Their work gives an unforgeability notion (Def. 9) for signatures with (perfectly) re-randomizable keys. The new unforgeability notion requires it to be infeasible for an adversary to output a forgery under either the original or a re-randomized key, even when they control the randomness. The authors note, however, that security does not trivially follow from the “standard” regular notion of (existential) unforgeability. In fact, schemes as the one from Boneh and Boyen [BB04] or Camenisch and Lysyanskaya [CL04] are insecure w.r.t. this stronger notion although their keys can be randomized. Nevertheless, Fleischhacker *et al.* give two constructions fulfilling their security notion. The first one is in the ROM and is a somewhat folklore variant of Schnorr [Sch90], also discussed in [DS19] and [BCG⁺20]. The second one is secure in the standard model and is a variant of a scheme given by Hofheinz-Kiltz [HK08, HK12]. Bowe *et al.* [BCG⁺20] refer to the Schnorr variant as a *randomizable signature* and provide a similar formalization for the unforgeability property under the name of *existential unforgeability under randomization*.

The difference is that the adversary in [BCG⁺20] can only obtain signatures for the initial secret key and not from re-randomized ones. Therefore, the unforgeability notion presented in [BCG⁺20] is strictly weaker than the one from [FKM⁺16].

Definition 9 (Unforgeability under Re-Randomized Keys [FKM⁺16]). A signature scheme Γ under re-randomizable keys is *unforgeable* if, for every security parameter λ , message m and key randomizer ρ , the advantage of any PPT algorithm $\mathcal{A}$ defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{UNF}}(\lambda) := \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNF}}(\lambda) \Rightarrow \text{true}] \leq \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNF}}(\lambda)$ is shown in Fig. 4.

The authors of [FKM⁺16] aim to construct so called unlinkable sanitizable signatures [BFF⁺09, BFLS10] and thus require unlinkability on re-randomized keys. We note that this unlinkability is already baked into their correctness notion, requiring that, for uniform randomness, re-randomized and fresh keys are identically distributed. Bowe *et al.* [BCG⁺20] introduce an explicit unlinkability notion that they consider a “computational relaxation” of the previous one from [FKM⁺16]. Their definition (Def. 10) is close to the *class-hiding* notion from [CL19] but weaker since the adversary is only able to get signatures from one of the keys. Furthermore, Bowe *et al.* define re-randomizable signatures as providing existential unforgeability and a property called *injective randomization*. The latter states that obtaining the same randomized key for two different key randomizers is impossible.

Definition 10 (Unlinkability [BCG⁺20]). A signature scheme Γ is *unlinkable* if, for every security parameter λ and key randomizer ρ , the advantage of any PPT algorithm $\mathcal{A}$ defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$ is shown in Fig. 5.

3.3 Signatures with Key Blinding

Eaton *et al.* [ESS21] used the term *key blinding* to describe signature schemes for which the public key can be randomized (*i.e.*, masked or blinded, in their terminology). Similar to other works, given two randomized public keys and associated key randomizers, there should be no way to tell if they were generated from the same initial public key or not without knowledge of such key.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$	Oracle $\text{Sign}(\text{sk}, m)$
$\text{pp} \leftarrow \text{PPGen}(1^\lambda); b \leftarrow \{0, 1\}; \rho \leftarrow \mathcal{KR}; (\text{sk}_1, \text{pk}_1) \leftarrow \text{KGen}(\text{pp})$ $(\text{sk}_2^0, \text{pk}_2^0) \leftarrow \text{KGen}(\text{pp}); m \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}_1, \cdot)}(\text{pp}, \text{pk}_1)$ $\text{pk}_2^1 \leftarrow \text{RandPK}(\text{pk}_1, \rho); \text{sk}_2^1 \leftarrow \text{RandSK}(\text{sk}_1, \rho); \sigma \leftarrow \text{Sign}(\text{sk}_2^b, m)$ $b' \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}_1, \cdot)}(\text{pk}_1, \sigma, \text{pk}_2^b); \text{return } b = b'$	$\text{return } \text{Sign}(\text{sk}, m)$

Fig. 5. Unlinkability experiment from [BCG⁺20].

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$	
$\Sigma \leftarrow \emptyset; b \leftarrow \{0, 1\}; \text{pp} \leftarrow \text{PPGen}(1^\lambda); \{\text{sk}_i, \text{pk}_i\}_{i \in \{0, 1\}} \leftarrow \text{KGen}(\text{pp})$ $(\rho^*, \text{st}) \leftarrow \mathcal{A}_1^{\text{RandPK}_1(\cdot), \text{Adapt}_1(\cdot, \cdot)}(1^\lambda); \text{pk}^* \leftarrow \text{RandPK}(\text{pk}_b, \rho^*); \Sigma \leftarrow \Sigma \cup \{(\text{pk}^*, \rho^*)\}$ $b' \leftarrow \mathcal{A}_2^{\text{RandPK}_2(\cdot), \text{Adapt}_2(\cdot, \cdot)}(\text{pk}^*); \text{return } b = b'$	
Oracle $\text{RandPK}_1(\rho)$	Oracle $\text{RandPK}_2(\rho)$
$\text{pk}' \leftarrow \text{RandPK}(\text{pk}_1, \rho)$ $\Sigma \leftarrow \Sigma \cup \{(\text{pk}', \rho)\}$ $\text{return } \text{pk}'$	$\text{if } \rho = \rho^* \text{ then return } \text{pk}^*$ $\text{else } \text{pk}' \leftarrow \text{RandPK}(\text{pk}_1, \rho); \Sigma \leftarrow \Sigma \cup \{(\text{pk}', \rho)\}$ $\text{return } \text{pk}'$
Oracle $\text{Adapt}_1(m, \rho)$	
$\text{if } \rho \notin \Sigma \text{ return } \perp \text{ else } (\sigma, \cdot) \leftarrow \text{Adapt}(m, \text{Sign}(\text{sk}_1, m), \rho, \text{pk}_1); \text{return } \sigma$	
Oracle $\text{Adapt}_2(m, \rho)$	
$\text{if } \rho \notin \Sigma \text{ return } \perp; \text{if } \rho = \rho^* \text{ then } (\sigma, \cdot) \leftarrow \text{Adapt}(m, \text{Sign}(\text{sk}_b, m), \rho, \text{pk}_b); \text{return } \sigma$ $\text{else } (\sigma, \cdot) \leftarrow \text{Adapt}(m, \text{Sign}(\text{sk}_1, m), \rho, \text{pk}_1); \text{return } \sigma$	

Fig. 6. Unlinkability experiment from [ESS21].

Signatures with key blinding must satisfy two security requirements: unforgeability and unlinkability. For the former, they rely on the “standard” unforgeability notion with the caveat that an adversary has to provide a tuple (ρ, σ, m) instead of (m, σ, pk) for verification to succeed. For the latter, they introduce a notion of unlinkability for signatures with key blinding (Def. 11). This notion captures the fact that even if an adversary has access to the RandPK and Adapt oracles, they still cannot distinguish a randomized key created from the original public key from one randomized from a random key. They introduce a property called *independent blinding*, which asks that the distribution of the output of the blinding function is independent of its input (essentially, it is non-deterministic). Hence, even if an adversary sees n randomizations from a public key, they will learn no information about the original public key.

Definition 11 (Unlinkability [ESS21]). *A signature scheme Γ is unlinkable if, for every security parameter λ , message m and key randomizer ρ , the advantage of any PPT adversary $\mathcal{A}$ defined by $\text{Adv}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$ is shown in Fig. 6.*

More recently, Eaton *et al.* [ELW23] extended the study of signatures with key-blinding in an attempt to capture additional applications ([ESS21] only considered anonymity networks). The unlinkability property from [ELW23] differs slightly from the previous one and states that *an adversary without knowledge of the long-term public key but who observes many blinded public keys and signatures that verify under those blinded public keys cannot distinguish between a blinding key of the long-term public key or a blinding of a freshly generated public key*. Compared with [ESS21], this formalization (Def. 12) does not allow the adversary to query the RandPK oracle for a chosen ρ , which allows them to receive a corresponding randomized public key. The authors’ interest in this modified notion of unlinkability is that it allows for a key blinding scheme that admits an unblinding functionality. This can be necessary for certain applications where only trusted parties execute the unblinding process. Hence, they treat the ρ as privileged information unavailable to the adversary. However, this property can be considered weaker as the adversary is restricted from learning ρ . We also note that they call this unlinkability with unblinding property “bidirectional” blinding. This contrasts with the notion of “one-way” blinding from [ESS21] where no unblinding is supported.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$

$\Sigma \leftarrow \emptyset; b \leftarrow \mathbb{S} \{0, 1\}; \text{pp} \leftarrow \mathbb{S} \text{PPGen}(1^\lambda); \{\text{sk}_i, \text{pk}_i\}_{i \in \{0, 1\}} \leftarrow \mathbb{S} \text{KGen}(\text{pp}); \rho^* \leftarrow \mathbb{S} \mathcal{KR}$
 $\text{pk}^* \leftarrow \text{RandPK}(\text{pk}_b, \rho^*); \Sigma \leftarrow \Sigma \cup \{(\text{pk}^*, \rho^*)\}; b' \leftarrow \mathbb{S} \mathcal{A}^{\text{RandPK}(\cdot), \text{Adapt}(\cdot, \cdot)}(\text{pk}^*)$
return $b = b'$
 Oracle $\text{RandPK}() : \rho' \leftarrow \mathbb{S} \mathcal{KR}; \text{pk}' \leftarrow \text{RandPK}(\text{pk}_1, \rho'); \Sigma \leftarrow \Sigma \cup \{(\text{pk}', \rho')\};$ **return** pk'
 Oracle $\text{Adapt}(m, \text{pk})$
if $\text{pk} \notin \Sigma$ **return** $\perp$; **if** $\text{pk} = \text{pk}^*$ **then** $(\sigma, \cdot) \leftarrow \text{Adapt}(m, \text{Sign}(\text{sk}_b, m), \rho, \text{pk}_b)$
else $\rho^* \leftarrow \Sigma(\text{pk}^*); (\sigma, \cdot) \leftarrow \text{Adapt}(m, \text{Sign}(\text{sk}_1, m), \rho^*, \text{pk}_1);$ **return** σ

Fig. 7. Unlinkability experiment from [ELW23].

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{SUNF}}(\lambda)$

$\Sigma \leftarrow \emptyset; \text{pp} \leftarrow \mathbb{S} \text{PPGen}(1^\lambda); (\text{sk}, \text{pk}) \leftarrow \mathbb{S} \text{KGen}(\text{pp}); (\rho^*, m^*, \sigma^*) \leftarrow \mathbb{S} \mathcal{A}^{\text{Sign}(\text{sk}, \cdot, \cdot)}(\text{pk})$
if $\rho^* = \perp$ **then** $\text{pk}^* \leftarrow \text{pk}$ **else** $\text{pk}^* \leftarrow \text{RandPK}(\text{pk}, \rho^*)$
return $\text{Verify}(m^*, \sigma^*, \text{pk}^*) \wedge (\rho^*, m^*, \sigma^*) \notin \Sigma$
 Oracle $\text{Sign}(\text{sk}, m, \rho)$
 $\Sigma \leftarrow \Sigma \cup \{(\rho, m, \sigma)\};$ **if** $\rho = \perp$ **return** $\text{Sign}(\text{sk}, m);$ **return** $\text{Sign}(\text{RandSK}(\text{sk}, \rho), m)$

Fig. 8. Unforgeability experiment from [ELW23].

Definition 12 (Unlinkability [ELW23]). A signature scheme Γ is unlinkable if, for every security parameter λ , message m and key randomizer ρ , the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNL}}(\lambda)$ is shown in Fig. 7.

Both works base their techniques on two conditions: (1) an adversary with access to a blinding oracle and signing oracle cannot distinguish between a new blinding of a long-term key and a blinding of a freshly-chosen key (the blinded public keys is independent of the long-term public key), and (2) signatures with an identical distribution that are produced from blinded public keys depend only on the blinded public key and not on long-term public key (signatures leak no information about the long-term public key). Nevertheless, the two scenarios are quite different and talking about unlinkability in both cases may be confusing.

The unlinkability notion from [ESS21] considers an adversary who aims to *extract* the long-term public key when confronted with (possibly many) randomizations of it for which they know the key randomizer (*i.e.*, blinding factor). However, one can also consider an adversary who knows the long-term public key and aims to link it with a randomized public key without knowledge of the key randomizer (as in the previous unlinkability notions). For this reason, we propose a different formalization in Sec. 4 (Def. 17), to better capture this issue.

The work of [ELW23] also introduces a notion of *strong unforgeability* (Def. 13). A weaker notion of this property only allows the adversary to modify σ^* , which would allow for forged signatures to be valid under any ρ .

Definition 13 (Strong Unforgeability [ELW23]). A signature scheme Γ is strongly unforgeable if, for every security parameter λ , message m and key randomizer ρ , the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{SUNF}}(\lambda) := \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{SUNF}}(\lambda) \Rightarrow \text{true}] \leq \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{SUNF}}(\lambda)$ is shown in Fig. 8.

3.4 Signatures with honestly randomized keys

Deterministic wallets (further discussed in Sec. 5.4) require a signature scheme with certain properties as outlined in [DFL19]. For example, they require a perfect randomization property equivalent to that of [CL19]. Further, they require *unforgeability under honestly rerandomized keys* (Def. 14), which states that an adversary cannot produce a forgery as long as keys are correctly randomized.

Definition 14 (Unforgeability under honestly rerandomized keys [DFL19]). A signature scheme Γ has unforgeability under honestly rerandomized keys if, for every security parameter λ , message m and key randomizer ρ , the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathbf{Adv}_{\Gamma, \mathcal{A}}^{\text{UNF-hrk}}(\lambda) := \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNF-hrk}}(\lambda) \Rightarrow \text{true}] \leq \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{UNF-hrk}}(\lambda)$ is shown in Fig. 9.

Experiment $\mathbf{Exp}_{\mathcal{F}, \mathcal{A}}^{\text{UNF-hrk}}(\lambda)$	
$\Sigma_1 \leftarrow \emptyset; \Sigma_2 \leftarrow \emptyset; (\text{sk}, \text{pk}) \leftarrow_{\$} \text{KGen}(1^\lambda); (m^*, \sigma^*, \rho^*) \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}, \cdot, \cdot), \text{Rand}(\cdot)}(\text{pk})$	
$\text{pk}^* \leftarrow \text{RandPK}(\text{pk}, \rho^*); \text{return } m^* \notin \Sigma_1 \wedge \rho^* \in \Sigma_2 \wedge \text{Verify}(\text{pk}^*, \sigma^*, m^*)$	
Oracle $\text{Sign}(\text{sk}, m, \rho)$	Oracle $\text{Rand}()$
If $\rho \notin \Sigma_2$, return $\perp$; $\text{sk}' \leftarrow \text{RandSK}(\text{sk}, \rho)$	$\rho \leftarrow_{\$} \mathcal{KR}; \Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\}; \text{return } \rho$
$\sigma \leftarrow \text{Sign}(\text{sk}', m); \Sigma_1 \leftarrow \Sigma_1 \cup \{m\}; \text{return } \sigma$	

Fig. 9. Unforgeability under honestly rerandomized keys experiment from [DFL19].

Def. 14 is strictly weaker than Def. 9 from Sec. 3.2, since the adversary does not control the randomness used to convert the challenge key. Furthermore, the adversary is only allowed to query the signing oracle for randomizations of the secret key but not the secret key itself.

3.5 Updatable signatures

Updatable signatures [CRS⁺21] are the signature equivalent to updatable encryption (UE) [BLMR13], i.e., the main motivation is periodical key-rotation. They work with the concept of epochs (keys are updated in every new epoch) and require an *update token* ρ_{e+1} to move (i.e., adapt) signatures produced under a key in epoch e to signatures valid under the key in the next epoch $e+1$. Cini *et al.* [CRS⁺21] consider constructions from key-homomorphic signatures [DS19] as well as dedicated ones. Updatable signatures include a function, **Next**, which can be thought of as being a bundled function for **RandPK** and **RandSK** that samples its own key randomizer. We recall their unforgeability notion in the full version [CGH⁺23] (Appendix C). In the game, the adversary can use the oracles to obtain signatures, updates of signatures (**Update**), public keys (**Next**), secret keys (**CorruptKey**), and update tokens (**CorruptToken**), with the restriction that trivial forgeries are excluded. To capture them, the authors (in analogy to UE in [LT18]) create a recursively defined set of messages and epochs that the adversary must be able to compute by correctness (cf. [CRS⁺21] for details). In other words, the definition does not count forgeries if the adversary’s forgery is under a key on which they’ve either corrupted a signature from the previous key along with an update token or a signature from the next key along with an update token.

Cini *et al.* [CRS⁺21] also include an *unlinkable updates under chosen message attack* definition, which ensures that updated signatures are indistinguishable from fresh ones. This definition of unlinkability does not involve distinguishing the origin of a randomized public key, but instead, only distinguishing the origin of a signature (the origin being which value the challenge value was adapted from). Because of this, we also defer its presentation to our extended version. However, we note that this notion of unlinkability is implied by perfect adaption since the adversary is stronger in the adaptability game (no corruption or update oracles are given).

Kloof *et al.* [KLR19] construct a one-time signature (instantiated based on the one-time SPS from [KPW15]) with an updatable signature notion to provide integrity in UE. Their construction is only proven secure under non-randomizable keys and is a one-time signature. Whether their approach can be adapted to provide security for randomizable keys and sign multiple messages without leading to forgeries (perhaps using the compiler from [KPW15]) remains open.

4 Systematization

In this section, we propose unifying definitions to capture all the relevant properties of SWRK: *unforgeability*, *unlinkability* and *unextractability*. Subsequently, we classify the existing constructions.

4.1 Unforgeability

We opt to merge the definitions from [FKM⁺16] and [DFL19] (Def. 9 and Def. 14) to consider two cases: when the key randomizer is honestly generated and when the adversary can arbitrarily pick it. To reflect this, we parametrize the definition (Def. 15) by α , where $\alpha = 0$ means the key randomizer is honestly generated.

Observe that the forgery should be with respect to the original key pair. Otherwise, the adversary could run **KGen** to obtain a random key pair, produce a signature, randomize it, and present

Experiment $\text{Exp}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UNF}}(\lambda)$ $\Sigma_1 \leftarrow \emptyset; \Sigma_2 \leftarrow \emptyset; \text{pp} \leftarrow_{\\$} \text{PPGen}(1^\lambda); (\text{sk}, \text{pk}) \leftarrow_{\\$} \text{KGen}(\text{pp})$ $(m^*, \sigma^*, \rho^*) \leftarrow_{\\$} \mathcal{A}^{\text{Sign}(\text{sk}, \cdot, \cdot), \text{Rand}(\cdot)}(\text{pk}); \text{pk}^* \leftarrow \text{RandPK}(\text{pk}, \rho^*)$ return $(m^*, \boxed{\sigma^*}) \notin \Sigma_1 \wedge (\rho^* \in \Sigma_2 \vee \alpha = 1) \wedge \text{Verify}(m^*, \sigma^*, \text{pk}^*)$	Oracle $\text{Sign}(\text{sk}, m, \rho)$ if $\rho \notin \Sigma_2 \wedge \alpha = 0$ return $\perp$ if $\rho = \perp$ $\sigma \leftarrow_{\\$} \text{Sign}(\text{sk}, m)$ else $\sigma \leftarrow \text{Sign}(\text{RandSK}(\text{sk}, \rho), m)$ $\Sigma_1 \leftarrow \Sigma_1 \cup \{(m, \boxed{\sigma})\};$ return σ
	Oracle $\text{Rand}(\cdot)$ $\rho \leftarrow_{\\$} \mathcal{KR}$ $\Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\}$ return ρ

Fig. 10. Our α -unforgeability experiment. Solid boxes refer to *strong* unforgeability.

Experiment $\mathbf{Exp}_{\text{SWRK},A}^{(\alpha,\beta,\gamma)-\text{UNL}}(\lambda)$

st $\leftarrow \emptyset$; **b** $\leftarrow_{\$} \{0, 1\}$; $\rho \leftarrow_{\$} \mathcal{KR}$; **if** $\alpha = 0$ **then** $\text{pp} \leftarrow_{\$} \text{PPGen}(1^\lambda)$ **else** $(\text{pp}, \text{st}) \leftarrow \mathcal{A}_0(1^\lambda)$
if $\beta = 0$ **then** $\{\text{sk}_i, \text{pk}_i\}_{i \in \{0,1\}} \leftarrow \text{KGen}(\text{pp})$ **else** $(\{\text{sk}_i, \text{pk}_i\}_{i \in \{0,1\}}, \text{st}) \leftarrow \mathcal{A}_1(\text{st}, \text{pp})$
if $\exists i \in \{0, 1\}$ such that $\text{VerKey}(\text{sk}_i, \text{pk}_i) = 0$ **return** 0
 $\text{pk}' \leftarrow \text{RandPK}(\text{pk}_b, \rho)$; $\text{sk}' \leftarrow \text{RandSK}(\text{sk}_b, \rho)$
if $\gamma = \emptyset$ **then** $b' \leftarrow_{\$} \mathcal{A}_2(\text{pk}', \text{pk}_0, \text{pk}_1, \text{st})$ **else** $b' \leftarrow_{\$} \mathcal{A}_2^{\text{Sign}_\gamma(\cdot)}(\text{pk}', \text{pk}_0, \text{pk}_1, \text{st})$
return $b = b'$

Oracle $\text{Sign}_\gamma(m, \text{pk})$: **if** $(\text{pk}, \text{sk}) \notin K(\gamma)$ **return** $\perp$ **else** **return** $\text{Sign}(\text{sk}, m)$

Fig. 11. Our (α, β, γ) -unlinkability experiment.

$(m^*, \sigma^*, \rho^*, \text{pk}^*)$ to trivially win the game. As in [ELW23], our experiment also captures strong unforgeability. It is worth noting here that schemes supporting the adaptability of signatures cannot achieve strong unforgeability. In other words, those notions are mutually exclusive.

Definition 15 (α -Unforgeability). Let $\alpha \in \{0, 1\}$. A SWRK scheme is α -unforgeable if the advantage of any PPT adversary $\mathcal{A}$ defined by $\text{Adv}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UNF}}(\lambda) := \Pr[\text{Exp}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UNF}}(\lambda) \Rightarrow \text{true}] \leq \epsilon(\lambda)$, where $\text{Exp}_{\text{SWRK}, \mathcal{A}}^{\text{UNF}}(\lambda)$ is shown in Fig. 10.

4.2 Unlinkability

We propose a parametrized definition, (α, β, γ) -*unlinkability* (Def. 16), which is inspired by the notion of $(\mathcal{O}_1, \mathcal{O}_2, \alpha)$ -anonymity from [PS19]. In our case, we use $\alpha \in \{0, 1\}$ to denote whether or not the scheme is secure against adversarially chosen parameters. Similarly, $\beta \in \{0, 1\}$ denotes if the scheme is secure against adversarially chosen keys. Finally, the parameter γ denotes the set of keys for the signing oracle available to the adversary. For ease of exposition, we supply γ to a function K , resulting in the following scenarios to consider:

- $K(0) = \{\emptyset\}$: no signing oracle is available to the adversary.
- $K(1) = \{\{\text{sk}', \text{pk}'\}\}$: the signing oracle for the randomized key is available.
- $K(3) = \{\{\text{sk}', \text{pk}'\}, \{\{\text{sk}_i, \text{pk}_i\}_{i \in \{0,1\}}\}\}$: all signing oracles are available.

Definition 16 ((α, β, γ)-Unlinkability). Let $\alpha, \beta \in \{0, 1\}$ and let γ represent a set of keys parametrizing the signing oracle **Sign**. A SWRK scheme has (α, β, γ) -unlinkability if the advantage of any PPT adversary $\mathcal{A} = \{\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2\}$ defined by $\text{Adv}_{\text{SWRK}, \mathcal{A}}^{(\alpha, \beta, \gamma)\text{-UNL}}(\lambda) := 2 \cdot \Pr[\text{Exp}_{\text{SWRK}, \mathcal{A}}^{(\alpha, \beta, \gamma)\text{-UNL}}(\lambda) \Rightarrow \text{true}] - 1 = \epsilon(\lambda)$, where $\text{Exp}_{\text{SWRK}, \mathcal{A}}^{(\alpha, \beta, \gamma)\text{-UNL}}(\lambda)$ is shown in Fig. 11.

For schemes that support *perfect adaption* the adversary can run the **Adapt** algorithm by herself on the signatures obtained through the oracles. Thus, the adversary can locally compute signatures from randomized secret keys.

Definition 16 captures unlinkability for the following configurations: (0,0,0), (0,0,1), (0,0,3), (1,0,0), (1,0,1), (1,0,3), (0,1,0), (0,1,1), (1,1,0) and (1,1,1). We assume that the secret keys are not given to

Experiment $\mathbf{Exp}_{\text{SWRK},\mathcal{A}}^{(\alpha,\beta)\text{-UNE}}(\lambda)$

$\text{st} \leftarrow \emptyset; b \leftarrow \mathbb{S} \{0, 1\}; \text{if } \alpha = 0 \text{ then } \text{pp} \leftarrow \text{PPGen}(1^\lambda) \text{ else } (\text{pp}, \text{st}) \leftarrow \mathcal{A}_0(\text{st}, 1^\lambda)$
 $\{\text{sk}_i, \text{pk}_i\}_{i \in \{0,1\}} \leftarrow \mathbb{S} \text{KGen}(\text{pp}); \text{if } \beta = 0 \text{ then } \rho \leftarrow \mathbb{S} \mathcal{KR} \text{ else } (\rho, \text{st}) \leftarrow \mathcal{A}_1(\text{st}, \text{pp});$
 $\text{pk}'_b \leftarrow \text{RandPK}(\text{pk}_b, \rho); \text{sk}'_b \leftarrow \text{RandSK}(\text{sk}_b, \rho); b' \leftarrow \mathbb{S} \mathcal{A}_2^{\text{Sign}(\cdot, \cdot)}(\text{pk}'_b, \rho, \text{st}); \text{return } b = b'$

Oracle $\text{Sign}(m, \gamma)$

if $\gamma = b$ **return** $\text{Sign}(\text{sk}_b, m)$ **elseif** $\gamma = 1 - b$ **return** $\text{Sign}(\text{sk}_{1-b}, m)$
elseif $\gamma = \perp$ **return** $\text{Sign}(\text{RandSK}(\text{sk}_b, \rho), m)$ **else return** $\perp$

Fig. 12. Our (α, β) -unextractability experiment.

the adversary for configurations $(\cdot, 0, \cdot)$. However, as discussed in [BHKs18], the adversary could be given those keys or even the random coins used to generate the key pairs. In such a case, one gets an unlinkability notion, which we call unlinkability under *key leakage*. Knowledge of the secret key provides a strictly stronger notion than $(\cdot, 0, 3)$ but weaker than the case where the adversary can generate the keys. For simplicity, we denote this intermediate notion as $(\cdot, 0, 3^*)$ and stress that Def. 16 can easily be updated so that the adversary receives the secret keys or random coins.

4.3 Unextractability

As discussed in Sec. 3.3, in some scenarios (*e.g.*, anonymity networks, see Sec. 5.1 for more details) the adversary is given access to randomizations of the long-term public key and the key-randomizer, but not to the long-term public key. The security notion for this case differs from the usual unlinkability one that is used, *e.g.*, in the context of anonymous credentials, where the adversary knows the long-term public key and tries to identify its randomizations without knowledge of the key-randomizer. Since information given to the adversary differs in each setting, we see the former notion as orthogonal to the latter instead of weaker or opposed. In this regard, we stress that there are schemes supporting both properties and that it is the application in question that determines which property should be used depending on their security needs. Therefore, we introduce the distinct notion of (α, β) -unextractability (Def. 17), where $(\alpha, \beta) \in \{0, 1\}$. In our experiment, the adversary can arbitrarily pick the parameters (parametrized by α) and/or the key-randomizer (parametrized by β) —assuming the strongest variant— and is given an honestly randomized key for which they must determine the corresponding long-term public key. As previously mentioned, the adversary’s goal in the experiment is to determine the long-term public key given knowledge of a randomized key and the corresponding key-randomizer.

Definition 17 ((α, β)-Unextractability). Let $\alpha, \beta \in \{0, 1\}$. A SWRK scheme has (α, β) -unextractability if the advantage of any PPT adversary $\mathcal{A} = \{\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2\}$ defined by $\mathbf{Adv}_{\text{SWRK},\mathcal{A}}^{(\alpha,\beta)\text{-UNE}}(\lambda) := 2 \cdot \Pr \left[\mathbf{Exp}_{\text{SWRK},\mathcal{A}}^{(\alpha,\beta)\text{-UNE}}(\lambda) \Rightarrow \text{true} \right] - 1 = \epsilon(\lambda)$, where $\mathbf{Exp}_{\text{SWRK},\mathcal{A}}^{(\alpha,\beta)\text{-UNE}}(\lambda)$ is shown in Fig. 12.

4.4 Classification

An up-to-date classification of SWRK constructions based on our proposed formalization is given in Table 1. We include schemes that were not originally conceived as SWRK but that can easily be adapted (see Appendix B in [CGH⁺23]).

Remark 1 (On the relation between unlinkability and unextractability). The classification in Table 1 suggests that only schemes providing all signing oracles in the unlinkability experiment (*i.e.*, $\gamma = 3$) can achieve $(0, 1)$ -UNE unextractability. However, schemes achieving only $\gamma = 1$ unlinkability can be $(0, 1)$ -UNE secure. Consider the BLS signature scheme with a deterministic transformation T (*i.e.*, $\rho = \varepsilon$). The idea is to compute the transformed BLS key using the formula $\text{sk}' = \text{sk} \cdot \text{H}(\text{Sign}(\text{sk}, \text{"0"}), \text{pk})$. In other words, we randomize the secret key by multiplying it with the hash value of the pre-transformed public key and a signature under “0”. This transformation is correct since BLS signatures are unique, making the process deterministic without requiring the factor ρ . Unextractability holds because the adversary does not learn pk nor the randomizing factor $\text{H}(\text{Sign}(\text{sk}, \text{"0"}), \text{pk})$. This can be easily shown

Scheme	A	PA	α -UNF	(α, β, γ) -UNL	(α, β) -UNE	Setting	$ \mathbf{pk} $	$ \sigma $
Schnorr [Sch91, BCG ⁺ 20, FKM ⁺ 16]	✓	✗	1-UNF	$(1, 0, 3^*)$	✗	(EC)DL & ROM	$1 \mathbb{G} $	$1 \mathbb{Z}_p + 1 \mathbb{G} $
BLS [BLS04, DS19]	✓	✓	1-UNF	$(1, 1, 3)$	✗	BG ^{type-I} & ROM	$1 \mathbb{G} $	$1 \mathbb{G} $
Katz-Wang [KW03, GJKW07, DS19]	✓	✗	1-UNF	$(1, 0, 3)$	✗	(EC)DL & ROM	$2 \mathbb{G} $	$2 \mathbb{G} + 1 \mathbb{Z}_p $
Guillou-Quisquater [GQ90, DS19]	✓	✗	1-UNF	$(0, 1, 3)$	✗	RSA(CRS) & ROM	$1 \mathbb{Z}_N $	$1 \mathbb{Z}_e + 1 \mathbb{Z}_N ^{**}$
Waters [Wat05, BFG13, DS19]	✓	✓	1-UNF	$(1, 1, 3)$	✗	BG ^{type-III} & CDH	$1 \mathbb{G}_2 $	$1 \mathbb{G}_1 + 2 \mathbb{G}_2 $
Pointcheval-Sanders [PS16, DS19]	✓	✓	0-UNF	$(1, 1, 3)$	✗	BG ^{type-III} & GGM	$2 \mathbb{G}_2 $	$2 \mathbb{G}_1 $
AGOT [AGOT14, DS19]	✓	✓	0-UNF	$(1, 1, 3)$	✗	BG ^{type-I} & GGM	$2 \mathbb{G}_1 $	$2 \mathbb{G}_2 $
Ghadafi [Gha16, DS19]	✓	✓	0-UNF	$(1, 1, 3)$	✗	BG ^{type-III} & GGM	$2 \mathbb{G}_2 $	$3 \mathbb{G}_1 $
EdDSA [ELW23]	✗	✗	1-UNF [†]	$(1, 1, 3)$	$(0, 1)$ -UNE	(EC)DL	$1 \mathbb{G} $	$1 \mathbb{G} + 1 \mathbb{Z}_p $
ECDSA [ELW23]	✓	✗	1-UNF	$(1, 1, 3)$	$(0, 1)$ -UNE	(EC)DL	$1 \mathbb{G} $	$2 \mathbb{Z}_p $
Hofheinz-Kiltz [HK08, HK12, FKM ⁺ 16]	✗	✗	1-UNF	$(1, 1, 3)$	✗	BG & q -SDH	$1 \mathbb{G}_2 $	$1 \mathbb{Z}_p + 1 \mathbb{G}_1 $
BHKS1 (Scheme 4) [BHKS18]	✓	✓	1-UNF [‡]	$(1, 0, 3^*)$	✗	BG & DLIN & DDH	$(\ell + 5) \mathbb{G}_1 $	$2 \mathbb{G}_1 + 1 \mathbb{G}_2 $
BHKS2 (Scheme 5) [BHKS18]	✓	✓	1-UNF [‡]	$(0, 0, 3^*)$	✗	BG & DLIN & DDH	$3 \mathbb{G}_1 $	$2 \mathbb{G}_1 + 1 \mathbb{G}_2 $
Crites-Lysyanskaya [CL19]	✓	✓	1-UNF [‡]	$(1, 0, 3)$	✗	BG ^{type-III} & GGM	$\ell \mathbb{G}_1 $	$2 \mathbb{G}_1 + 1 \mathbb{G}_2 $
CLPK [CLPK22]	✓	✓	1-UNF [‡]	$(0, 0, 3)$	✗	BG ^{type-III} & CRS	$(2 + \ell) \mathbb{G}_2 $	$9 \mathbb{G}_1 + 4 \mathbb{G}_2 $
ESS (Dilithium) [ESS21]	✗	✗	1-UNF	$(1, 0, 3)$	$(0, 1)$ -UNE	MLWE	7.7kB	5.7kB
ESS (Picnic) [ESS21]	✗	✗	1-UNF	$(1, 0, 3)$	$(0, 1)$ -UNE	MPC-in-the-head	32B	$\approx 75\text{kB}^{***}$
ESS (LegRoast) [ESS21]	✗	✗	1-UNF	$(1, 0, 3)$	$(0, 1)$ -UNE	PRF	0.50kB	7.94kB
ESS (CSL-FiSh) [ESS21]	✗	✗	1-UNF	$(1, 0, 3)$	$(0, 1)$ -UNE	CSIDH	32B	1.8-2.1kB
US (Fig. 7) [CRS ⁺ 21]	✓	✓	1-UNF	$(1, 1, 3)$	✗	BG ^{type-I} & ROM	$1 \mathbb{G} $	$1 \mathbb{G} $
HRK (Fig. 10) [DFL19]	✗	✗	0-UNF	$(1, 1, 3)$	✗	(EC)DL	$1 \{0, 1\}^\lambda + 2 \mathbb{Z}_p $	$1 \mathbb{G} $

Table 1. Classification of SWRK schemes in terms of their adaption (A), perfect adaption (PA), unforgeability (α -UNF, where [†] refers to the strong variant and [‡] means equivalence classes), unlinkability $((\alpha, \beta, \gamma)$ -UNL, where $*$ means that keys can be leaked to the adversary) and unextractability $((\alpha, \beta)$ -UNE) properties. We also include the setting in which they work, and the size of $\mathbf{pk}$ (ℓ parametrizes the message vector’s length) and σ . ^{**} e is a large prime roughly the size of $\phi(N)$. ^{***} Estimated size.

under the DDH assumption in the ROM. On the other hand, the scheme cannot achieve unlinkability with $\gamma = 3$ because an adversary can query the signatures under both original public keys, compute the randomizing factors $H(\text{Sign}(\text{sk}_0, “0”), \mathbf{pk}_0)$ and $H(\text{Sign}(\text{sk}_1, “0”), \mathbf{pk}_1)$, and check which public key corresponds to $\mathbf{pk}'$.

5 Applications

5.1 Anonymity networks: Tor

Anonymity networks allow users to conceal their internet history from website operators, Internet Service Providers (ISPs), and any intermediaries in the network path. The most famous one is the Tor network alongside its *onion services*.

As defined by version 3 of Tor’s rendezvous specification [TTP23], the Ed25519 signature scheme [BDL⁺12] is used as a SWRK. Long-term keys in this signature scheme are made w.r.t. a generator G of a cyclic group of large prime size ℓ and are an integer $a \in [\ell - 1]$. The corresponding public key is $A = G^a$, which can be randomized using a nonce τ and hash function H to obtain $\rho \leftarrow H(\tau \| A)$ with $\rho \in [\ell - 1]$. The randomized key pair, $(\rho \cdot a \bmod \ell, A^\rho)$, is entirely fresh and compatible with Ed25519 for signing and verification (see Appendix A in [CGH⁺23]). The application requires the scheme to provide two essential properties: *unlinkability* and *unforgeability*. As discussed by [ESS21, Hop13], Tor’s functionality, when instantiated in their post-quantum setting, achieves 1-UNF and $(1, 0, 3)$ -UNL when seen as a SWRK. Furthermore, a randomized public key is treated as *completely public* in Tor and should leak no information about the long-term public key. Hence, the scheme should be at least $(0, 1)$ -UNE (*i.e.*, to provide “key-blinding without unblinding” following the terminology from [ELW23]).

5.2 Rate-limiting Privacy Pass

Privacy Pass [DGS⁺18] is a protocol that relies on challenges (*e.g.*, human attestations) to assess if a client is honest (*i.e.*, not fraudulent) to give certain amount of unlinkable and unforgeable tokens, to use in future interactions if the client is deemed honest (*i.e.*, they solve the challenge successfully). The goal is to reduce the number of challenges presented to a client as these can impact usability. This is particularly useful for clients who are assigned to IPs with poor reputations.

An extended protocol version [CDVW23, DIW23] includes a third party, dividing the functionalities into attestation and issuance. Clients interact with an attester and issuer service to produce tokens. A rate-limited version [HIP+23] extended this architecture with the ability for the attester to limit the number of tokens clients can request, but without the attester learning which services a specific client interacts with. The scheme requires key-randomization, with the attester’s ability to rollback the process. To prevent a dictionary attack, the protocol requires unlinkability, and unforgeability. As discussed by [ELW23], the functionality of rate-limiting Privacy Pass (as an example of a key-blinding signature scheme) achieves 1-UNF[†] (the strong notion is *only* provided in the ECDSA version) and (1, 1, 3)-UNL, given our definitions. Note that in Privacy Pass a randomized public key is treated as *private information* and never given to the adversary. [ELW23] cites this scheme as an example of “key-blinding with unblinding”.

5.3 Anonymous Credentials

Anonymous credentials (see Appendix A.3 in the full version for a background) are usually constructed with signatures on commitments to a user’s identity. To show a credential, the signature and commitment are randomized together and knowledge of the identity is proven. This prevents a malicious signer from linking a user’s credential to a particular signing. However, knowledge of the signer can reveal sufficient information to fully de-anonymize users in some scenarios: a problem that has been studied in different settings [CL19, CL21, BEK+21, CLPK22, CDLPK23, MBG+23a]. Potential solutions to this problem require different properties. In issuer-hiding credentials (*e.g.*, [BEK+21, CLPK22]), a verifier attempts to link a signature to a particular public key after the public key has been randomized. For this reason, signatures with strong unlinkability (*i.e.*, (1, 1, 3)-UNL) are desired. Unfortunately, known constructions of issuer-hiding credentials only provide (1, 0, 3)-UNL.

Another desired property is to allow the signer to delegate their power to other users. In *delegatable* anonymous credentials (DACs), a verifier often sees a chain of public keys (as in [CL19, Fuc11]). In this chain, the root is unrandomized and trusted by the verifier, the intermediate keys are those of delegators and the last key is from the user who performs the credential showing. This means that (1)-UNF is a desirable property. Without this property, each key in the chain would need an attached (randomizable) proof ensuring that it is computed correctly. To construct DACs directly from SWRK signatures, (0, 0, 3)-UNL is desired as intermediate signers will have to issue signatures from their randomized keys or else signers will reveal the origin of their own delegation chain [CL19]. Unextractability is not required for anonymous credentials as the randomization factor isn’t known to the verifier nor to the signer.

5.4 Deterministic Wallets & Stealth Addresses

Deterministic wallets tackle the problem of a user who wants to send money through a blockchain from many different transactions to their wallet without linking it to each transaction (see Appendix A.4 in [CGH+23] for hot/cold wallets). This can be achieved using different keys (where the user generates a new key for each transaction). However, this causes the number of keys to scale with the number of transactions. Deterministic wallets provide a single key pair, where the public key can be randomized so that a single wallet can have multiple unlinkable keys, thus requiring at least (0, 0, 3)-UNL as it needs to support randomized wallets that create further transactions (signatures). With deterministic wallets, we further need to assume the signature scheme is unforgeable for honestly randomized keys since the user will only send money to these honestly created wallets. This means that even if an adversary forges a signature for a randomized public key (if it was maliciously randomized), the wallet will have no value to steal. Thus, they can be realized with a signature scheme that achieves 0-UNF.

Stealth addresses techniques are aimed to generate one-time addresses for each transaction so that the privacy of cryptocurrencies is increased. The technique has many implementations: the ones using SWRK are among the most relevant. In them, the user can use RandPK on the recipient’s public key and send money to the randomized public key. The transaction is finalized with the user sending the randomizer to the recipient, allowing for the redemption of funds. Since the new address can be chosen maliciously, the stealth address application requires at least (0, 0, 3)-UNL and 1-UNF. As shown in the recent work by Pu *et al.* [PTDH23], stealth addresses are closely related to asynchronous remote key generation for FIDO tokens [FGK+20]. In this application, the primary device registers randomized public keys in the name of the backup device (*e.g.*, stored in a safe). In case the primary

device is lost, the backup device can restore the secret key and successfully authenticate to the server while leveraging the registration which is solely executed with the primary device. Both devices use a key agreement protocol to generate a shared randomizer, allowing the backup device to recompute it later. This application requires at least $(0, 0, 3)$ -UNL and 1-UNF, since the primary device could potentially be malicious.

5.5 Stronger Security for NIZK Proofs

As shown in [DS19], key-homomorphic signatures can be used to generically lift non-interactive witness indistinguishable and zero-knowledge proofs that provide soundness (or knowledge soundness) to ones that provide the stronger notion of simulation soundness (or simulation extractability). The basic idea is to extend a proof for a language $\mathcal{L}$ to a language $\mathcal{L} \vee \mathcal{L}_{key}$ and to add a public key of a signature scheme into the CRS. Loosely speaking, during proof computation one signs the proof with a secret key corresponding to a freshly sampled signature key pair. To prove the language $\mathcal{L}_{key}$, one proves that one knows a key randomizer that converts the fresh public key to the one in the CRS. This can only be done by the simulator knowing the trapdoor of the CRS (*i.e.*, the corresponding signing key). This application requires at least $(0, 0, 0)$ -UNL, 1-UNF and signature adaption. In [ARS20, AGRS24], the approach has been adapted to updatable signatures that provide (black-box) extractability (not to be confused with Def.17) *i.e.*, to make the key randomizer (black-box) extractable. These features can be used in generic constructions inspired by the aforementioned in [DS19] to build zk-SNARKs and circuit-succinct NIZK proofs with an updatable CRS [GKM⁺18].

6 Conclusions

We proposed a general framework to analyze the security of SWRK, providing an up-to-date literature review. Our definitions offer a flexible yet formal way to fully characterize signature constructions. They can help implementers to better identify the properties captured by a given scheme, helping avoid security risks. Moreover, our framework constitutes a step forward for standardization efforts in the area as it provides fine-grained separations between each notion. To justify our approach, we discussed practical applications for which such distinctions are highly relevant and classified all known constructions.

For future work a first direction is to explore further applications that might be enabled or benefit from this concept. A second direction is devising schemes with stronger properties for existing applications. In some cases, applications are realized with *weaker* properties but could benefit from stronger guarantees. For instance, anonymous credentials will benefit from signature constructions that achieve stronger unlinkability, such as $(0, 1, 3)$ -UNL or $(1, 1, 3)$ -UNL. A third and very important direction are post-quantum constructions: there are already some first works available [ADE⁺20, ESS21, Hu23, DEF⁺23, DEMS24]. However, they only focus on re-randomizing keys, but do not consider adaption. When it comes to signature adaption, the noisy nature of lattice-based cryptography seems to introduce non-trivial problems to overcome. Also, equivalence class signatures, *e.g.*, mercurial signatures, or also SFPK, from post-quantum assumptions are a completely unexplored field. Finally, while many of these schemes have practical applications, few of them have concrete implementations, and it is the subject of future work to properly implement them.

Acknowledgements. Daniel Slamanig was supported by the European Union through the Horizon Europe research programme SUNRISE (GA No. 101073821) and by the Austrian Science Fund (FWF) and netidee SCIENCE PROFET (GA No. P31621-N38). His work was done while being affiliated with AIT Austrian Institute of Technology. Scott Griffy was supported by NSF Award 2154170 as well as funding from the Peter G. Peterson Foundation.

References

- ABC⁺12. Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, abhi shelat, and Brent Waters. Computing on authenticated data. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 1–20. Springer, Heidelberg, March 2012.

- ADE⁺20. Nabil Alkeilani Alkadri, Poulami Das, Andreas Erwig, Sebastian Faust, Juliane Krämer, Siavash Riahi, and Patrick Struck. Deterministic wallets in a quantum world. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1017–1031. ACM Press, November 2020.
- AGOT14. Masayuki Abe, Jens Groth, Miyako Ohkubo, and Mehdi Tibouchi. Structure-preserving signatures from type II pairings. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 390–407. Springer, Heidelberg, August 2014.
- AGRS24. Behzad Abdolmaleki, Noemi Glaeser, Sebastian Ramacher, and Daniel Slamanig. Circuit-Succinct Universally-Composable NIZKs with Updatable CRS. In *2024 IEEE 37th Computer Security Foundations Symposium (CSF)*, pages 527–542, Los Alamitos, CA, USA, July 2024. IEEE Computer Society.
- ALP12. Nuttapon Attrapadung, Benoît Libert, and Thomas Peters. Computing on authenticated data: New privacy definitions and constructions. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 367–385. Springer, Heidelberg, December 2012.
- ARS20. Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. Lift-and-shift: Obtaining simulation extractable subversion and updatable SNARKs generically. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1987–2005. ACM Press, November 2020.
- BB04. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, Heidelberg, May 2004.
- BB18. Johannes Blömer and Jan Bobolz. Delegatable attribute-based anonymous credentials from dynamically malleable signatures. In Bart Preneel and Frederik Vercauteren, editors, *ACNS 18*, volume 10892 of *LNCS*, pages 221–239. Springer, Heidelberg, July 2018.
- BBD⁺10. Christina Brzuska, Heike Busch, Özgür Dagdelen, Marc Fischlin, Martin Franz, Stefan Katzenbeisser, Mark Manulis, Cristina Onete, Andreas Peter, Bertram Poettering, and Dominique Schröder. Redactable signatures for tree-structured data: Definitions and constructions. In Jianying Zhou and Moti Yung, editors, *ACNS 10*, volume 6123 of *LNCS*, pages 87–104. Springer, Heidelberg, June 2010.
- BCC⁺09. Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, Heidelberg, August 2009.
- BCG⁺20. Sean Bowe, Alessandro Chiesa, Matthew Green, Ian Miers, Pratyush Mishra, and Howard Wu. ZEXE: Enabling decentralized private computation. In *2020 IEEE Symposium on Security and Privacy*, pages 947–964. IEEE Computer Society Press, May 2020.
- BCM11. Mihir Bellare, David Cash, and Rachel Miller. Cryptography secure against related-key attacks and tampering. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 486–503. Springer, Heidelberg, December 2011.
- BDL⁺12. Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2(2):77–89, September 2012.
- BEK⁺21. Jan Bobolz, Fabian Eidens, Stephan Krenn, Sebastian Ramacher, and Kai Samelin. Issuer-hiding attribute-based credentials. In Mauro Conti, Marc Stevens, and Stephan Krenn, editors, *CANS 21*, volume 13099 of *LNCS*, pages 158–178. Springer, Heidelberg, December 2021.
- BF11. Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 149–168. Springer, Heidelberg, May 2011.
- BFF⁺09. Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of sanitizable signatures revisited. In Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009*, volume 5443 of *LNCS*, pages 317–336. Springer, Heidelberg, March 2009.
- BFG13. David Bernhard, Georg Fuchsbauer, and Essam Ghadafi. Efficient signatures of knowledge and DAA in the standard model. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 518–533. Springer, Heidelberg, June 2013.
- BFKW09. Dan Boneh, David Freeman, Jonathan Katz, and Brent Waters. Signing a linear subspace: Signature schemes for network coding. In Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009*, volume 5443 of *LNCS*, pages 68–87. Springer, Heidelberg, March 2009.
- BFLS10. Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schröder. Unlinkability of sanitizable signatures. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 444–461. Springer, Heidelberg, May 2010.
- BGG⁺14. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald,

- editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Heidelberg, May 2014.
- BHKS18. Michael Backes, Lucjan Hanzlik, Kamil Klucznik, and Jonas Schneider. Signatures with flexible public key: Introducing equivalence classes for public keys. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 405–434. Springer, Heidelberg, December 2018.
- BHSB19. Michael Backes, Lucjan Hanzlik, and Jonas Schneider-Bensch. Membership privacy for fully dynamic group signatures. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2181–2198. ACM Press, November 2019.
- BKM09. Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *Journal of Cryptology*, 22(1):114–138, January 2009.
- BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Heidelberg, August 2013.
- BLS04. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.
- BM99. Mihir Bellare and Sara K. Miner. A forward-secure digital signature scheme. In Michael J. Wiener, editor, *CRYPTO’99*, volume 1666 of *LNCS*, pages 431–448. Springer, Heidelberg, August 1999.
- BPT12. Mihir Bellare, Kenneth G. Paterson, and Susan Thomson. RKA security beyond the linear barrier: IBE, encryption and signatures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 331–348. Springer, Heidelberg, December 2012.
- CDD17. Jan Camenisch, Manu Drijvers, and Maria Dubovitskaya. Practical UC-secure delegatable credentials with attributes and their application to blockchain. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 683–699. ACM Press, October / November 2017.
- CDHK15. Jan Camenisch, Maria Dubovitskaya, Kristiyan Haralambiev, and Markulf Kohlweiss. Composable and modular anonymous credentials: Definitions and practical constructions. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 262–288. Springer, Heidelberg, November / December 2015.
- CDL⁺13. Jan Camenisch, Maria Dubovitskaya, Anja Lehmann, Gregory Neven, Christian Paquin, and Franz-Stefan Preiss. Concepts and languages for privacy-preserving attribute-based authentication. In Simone Fischer-Hübner, Elisabeth de Leeuw, and Chris Mitchell, editors, *Policies and Research in Identity Management*, pages 34–52, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
- CDLPK23. Aisling Connolly, Jérôme Deschamps, Pascal Lafourcade, and Octavio Perez Kempner. Protego: Efficient, revocable and auditable anonymous credentials with applications to hyperledger fabric. In *Progress in Cryptology – INDOCRYPT 2022: 23rd International Conference on Cryptology in India, Kolkata, India, December 11–14, 2022, Proceedings*, page 249–271, Berlin, Heidelberg, 2023. Springer-Verlag.
- CDVW23. Sofia Celi, Alex Davidson, Steven Valdez, and Christopher Wood. Privacy pass issuance protocol, 2023.
- CFW14. Dario Catalano, Dario Fiore, and Bogdan Warinschi. Homomorphic signatures with efficient verification for polynomial functions. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 371–389. Springer, Heidelberg, August 2014.
- CGH⁺23. Sofia Celi, Scott Griffy, Lucjan Hanzlik, Octavio Perez Kempner, and Daniel Slamanig. Sok: Signatures with randomizable keys. Cryptology ePrint Archive, Paper 2023/1524, 2023. <https://eprint.iacr.org/2023/1524>.
- CGH⁺24. Sofia Celi, Scott Griffy, Lucjan Hanzlik, Octavio Perez-Kempner, and Daniel Slamanig. Sok: Signatures with randomizable keys. In Jeremy Clark and Elaine Shi, editors, *Financial Cryptography and Data Security - 28th International Conference, FC 2024, Willemstad, Curaçao, March 4-8, 2024, Revised Selected Papers, Part II*, volume 14745 of *Lecture Notes in Computer Science*, pages 160–187. Springer, 2024.
- CGT23. Alishah Chator, Matthew Green, and Pratyush Ranjan Tiwari. Sok: Privacy-preserving signatures. *IACR Cryptol. ePrint Arch.*, page 1039, 2023.
- Cha85. David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Commun. ACM*, 28(10):1030–1044, oct 1985.
- CKLM14. Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Malleable signatures: New definitions and delegatable anonymous credentials. In Anupam Datta and Cedric Fournet, editors, *CSF 2014 Computer Security Foundations Symposium*, pages 199–213. IEEE Computer Society Press, 2014.
- CL01. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001.

- CL03. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289. Springer, Heidelberg, September 2003.
- CL04. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, August 2004.
- CL19. Elizabeth C. Crites and Anna Lysyanskaya. Delegatable anonymous credentials from mercurial signatures. In Mitsuru Matsui, editor, *CT-RSA 2019*, volume 11405 of *LNCS*, pages 535–555. Springer, Heidelberg, March 2019.
- CL21. Elizabeth C. Crites and Anna Lysyanskaya. Mercurial signatures for variable-length messages. *PoPETs*, 2021(4):441–463, October 2021.
- CLPK22. Aisling Connolly, Pascal Lafourcade, and Octavio Perez-Kempner. Improved constructions of anonymous credentials from structure-preserving signatures on equivalence classes. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 409–438. Springer, Heidelberg, March 2022.
- CRS⁺21. Valerio Cini, Sebastian Ramacher, Daniel Slamanig, Christoph Striecks, and Erkan Tairi. Updatable signatures and message authentication codes. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 691–723. Springer, Heidelberg, May 2021.
- DDH⁺15. Denise Demirel, David Derler, Christian Hanser, Henrich Pöhls, Daniel Slamanig, and Giulia Traverso. Prismacloud d4.4: Overview of functional and malleable signature schemes. 2015.
- DEF⁺23. Poulami Das, Andreas Erwig, Sebastian Faust, Julian Loss, and Siavash Riahi. BIP32-compatible threshold wallets. Cryptology ePrint Archive, Report 2023/312, 2023. <https://eprint.iacr.org/2023/312>.
- DEMS24. Poulami Das, Andreas Erwig, Michael Meyer, and Patrick Struck. Efficient post-quantum secure deterministic threshold wallets from isogenies. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, ASIA CCS ’24, page 522–532, New York, NY, USA, 2024. Association for Computing Machinery.
- DFL19. Poulami Das, Sebastian Faust, and Julian Loss. A formal treatment of deterministic wallets. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 651–668. ACM Press, November 2019.
- DGS⁺18. Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, July 2018.
- DHRR22. Benjamin Dowling, Eduard Hauck, Doreen Riepel, and Paul Rösler. Strongly anonymous ratcheted key exchange. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part III*, volume 13793 of *LNCS*, pages 119–150. Springer, Heidelberg, December 2022.
- DIW23. Alex Davidson, Jana Iyengar, and Christopher Wood. The privacy pass architecture, 2023.
- DS16. David Derler and Daniel Slamanig. Key-homomorphic signatures and applications to multiparty signatures. Cryptology ePrint Archive, Report 2016/792, 2016. <https://eprint.iacr.org/2016/792>.
- DS19. David Derler and Daniel Slamanig. Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. *Designs, Codes and Cryptography*, 87(6):1373–1413, Jun 2019.
- ELW23. Edward Eaton, Tancrede Lepoint, and Christopher A. Wood. Security analysis of signature schemes with key blinding. Cryptology ePrint Archive, Paper 2023/380, 2023.
- ER22. Andreas Erwig and Siavash Riahi. Deterministic wallets for adaptor signatures. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022, Part II*, volume 13555 of *LNCS*, pages 487–506. Springer, Heidelberg, September 2022.
- ESS21. Edward Eaton, Douglas Stebila, and Roy Stracovsky. Post-quantum key-blinding for authentication in anonymity networks. In Patrick Longa and Carla Ràfols, editors, *LATINCRYPT 2021*, volume 12912 of *LNCS*, pages 67–87. Springer, Heidelberg, October 2021.
- FD02. Lucas C. Ferreira and Ricardo Dahab. Blinded-key signatures: Securing private keys embedded in mobile agents. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, SAC ’02, page 82–86, New York, NY, USA, 2002. Association for Computing Machinery.
- FD04. L.C. Ferreira and R. Dahab. Optimistic blinded-key signatures. In *IEEE First Symposium on Multi-Agent Security and Survivability*, 2004, pages 65–72, 2004.
- FGK⁺20. Nick Frymann, Daniel Gardham, Franziskus Kiefer, Emil Lundberg, Mark Manulis, and Dain Nilsson. Asynchronous remote key generation: An analysis of yubico’s proposal for W3C WebAuthn. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 939–954. ACM Press, November 2020.
- FHS15. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 233–253. Springer, Heidelberg, August 2015.

- FHS19. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546, April 2019.
- FKM⁺16. Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 301–330. Springer, Heidelberg, March 2016.
- Fuc11. Georg Fuchsbauer. Commuting signatures and verifiable encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 224–245. Springer, Heidelberg, May 2011.
- Gha16. Essam Ghadafi. Short structure-preserving signatures. In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 305–321. Springer, Heidelberg, February / March 2016.
- GJKW07. Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the Diffie-Hellman problems. *Journal of Cryptology*, 20(4):493–514, October 2007.
- GKM⁺18. Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, August 2018.
- GL24. Scott Griffy and Anna Lysyanskaya. PACIFIC: privacy-preserving automated contact tracing featuring integrity against cloning. *IACR Commun. Cryptol.*, 1(2):12, 2024.
- GQ90. Louis C. Guillou and Jean-Jacques Quisquater. A “paradoxical” identity-based signature scheme resulting from zero-knowledge. In Shafi Goldwasser, editor, *CRYPTO’88*, volume 403 of *LNCS*, pages 216–231. Springer, Heidelberg, August 1990.
- GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Heidelberg, April 2008.
- GVW15. Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 469–477. ACM Press, June 2015.
- HHH⁺08. Stuart Haber, Yasuo Hatano, Yoshinori Honda, William Horne, Kunihiko Miyazaki, Tomas Sander, Satoru Tezoku, and Danfeng Yao. Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In Masayuki Abe and Virgil Gligor, editors, *ASIACCS 08*, pages 353–362. ACM Press, March 2008.
- HIP⁺23. Scott Hendrickson, Jana Iyengar, Tommy Pauly, Steven Valdez, and Christopher Wood. Rate-limited token issuance protocol, 2023.
- HK08. Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 21–38. Springer, Heidelberg, August 2008.
- HK12. Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. *Journal of Cryptology*, 25(3):484–527, July 2012.
- Hop13. Nicholas Hopper. Proving security of tor’s hidden service identity blinding protocol, 2013.
- HS14. Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 491–511. Springer, Heidelberg, December 2014.
- HS21. Lucjan Hanzlik and Daniel Slamanig. With a little help from my friends: Constructing practical anonymous credentials. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 2004–2023. ACM Press, November 2021.
- Hu23. Mingxing Hu. Post-quantum secure deterministic wallet: Stateless, hot/cold setting, and more secure. Cryptology ePrint Archive, Report 2023/062, 2023. <https://eprint.iacr.org/2023/062>.
- JMM19. Daniel Jost, Ueli Maurer, and Marta Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 159–188. Springer, Heidelberg, May 2019.
- JMSW02. Robert Johnson, David Molnar, Dawn Xiaodong Song, and David Wagner. Homomorphic signature schemes. In Bart Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 244–262. Springer, Heidelberg, February 2002.
- JS18. Joseph Jaeger and Igors Stepanovs. Optimal channel security against fine-grained state compromise: The safety of messaging. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018.

- KLR19. Michael Klooß, Anja Lehmann, and Andy Rupp. (R)CCA secure updatable encryption with integrity protection. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 68–99. Springer, Heidelberg, May 2019.
- KMP16. Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 33–61. Springer, Heidelberg, August 2016.
- KMPQ23. Saqib A. Kakvi, Keith M. Martin, Colin Putman, and Elizabeth A. Quaglia. Sok: Anonymous credentials. In Felix Günther and Julia Hesse, editors, *Security Standardisation Research*, pages 129–151, Cham, 2023. Springer Nature Switzerland.
- KPW15. Eike Kiltz, Jiaxin Pan, and Hoeteck Wee. Structure-preserving signatures from standard assumptions, revisited. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 275–295. Springer, Heidelberg, August 2015.
- KSD19. Mojtaba Khalili, Daniel Slamanig, and Mohammad Dakhilalian. Structure-preserving signatures on equivalence classes from standard assumptions. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 63–93. Springer, Heidelberg, December 2019.
- KW03. Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 2003*, pages 155–164. ACM Press, October 2003.
- LRSW99. Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *SAC 1999*, volume 1758 of *LNCS*, pages 184–199. Springer, Heidelberg, August 1999.
- LT18. Anja Lehmann and Björn Tackmann. Updatable encryption with post-compromise security. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 685–716. Springer, Heidelberg, April / May 2018.
- MBG⁺23a. Omid Mir, Balthazar Bauer, Scott Griffy, Anna Lysyanskaya, and Daniel Slamanig. Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023*, pages 30–44. ACM, 2023.
- MBG⁺23b. Omid Mir, Balthazar Bauer, Scott Griffy, Anna Lysyanskaya, and Daniel Slamanig. Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. Cryptology ePrint Archive, Paper 2023/1016, 2023.
- MSBM23. Omid Mir, Daniel Slamanig, Balthazar Bauer, and René Mayrhofer. Practical delegatable anonymous credentials from equivalence class signatures. *Proc. Priv. Enhancing Technol.*, 2023(3):488–513, 2023.
- MSM⁺16. Hiraku Morita, Jacob C. N. Schuldt, Takahiro Matsuda, Goichiro Hanaoka, and Tetsu Iwata. On the security of the schnorr signature scheme and DSA against related-key attacks. In Soonhak Kwon and Aaram Yun, editors, *ICISC 15*, volume 9558 of *LNCS*, pages 20–35. Springer, Heidelberg, November 2016.
- PS16. David Pointcheval and Olivier Sanders. Short randomizable signatures. In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 111–126. Springer, Heidelberg, February / March 2016.
- PS19. Sunoo Park and Adam Sealfon. It wasn’t me! - Repudiability and claimability of ring signatures. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 159–190. Springer, Heidelberg, August 2019.
- PTDH23. Sihang Pu, Sri Aravinda Krishnan Thyagarajan, Nico Döttling, and Lucjan Hanzlik. Post quantum fuzzy stealth signatures and applications. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM SIGSAC CCS 2023*, pages 371–385. ACM, 2023.
- San20. Olivier Sanders. Efficient redactable signature and application to anonymous credentials. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 628–656. Springer, Heidelberg, May 2020.
- SBZ02. Ron Steinfeld, Laurence Bull, and Yuliang Zheng. Content extraction signatures. In Kwangjo Kim, editor, *ICISC 01*, volume 2288 of *LNCS*, pages 285–304. Springer, Heidelberg, December 2002.
- Sch90. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO’89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.
- Sch91. Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.
- TTP23. Inc The Tor Project. Tor rendezvous specification - version 3, 2023.
- Wat05. Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, Heidelberg, May 2005.

Appendix

A Applications: Detailed discussion

A.1 Anonymity networks

We recall the exposition from [ESS21]. At a high level, onion services work by uploading a three-hop path (defined as a circuit in Tor terminology) to a Tor node called the introduction point, where the path begins. Once a number of introduction points have been picked, the host builds a set of documents called “hidden service descriptors” (or “descriptors”) and uploads them to a group of Hidden Service Directory (HSDir) nodes. These documents list the hidden service’s current introduction points and describe how to contact the hidden service. Because of Tor’s layered encryption, the introduction point does not know where the onion service lives, only where the next node in the path lives. To connect to the onion service, a client uses the `.onion` address to find the introduction point, which will then direct their communication towards the onion service.

The `.onion` address is the long-lived EdDSA public key of the onion service (the long-term master identity key). Time in the Tor network is divided into periods: the period length is a consensus parameter, and the period number is the number of periods that have occurred since the Unix epoch. So, given the public key, the nonce τ , and the consensus parameters of the Tor network, the key randomizer ρ is computed by hashing together the public key, the nonce, the current period number, and some parameters of both the Tor network and the signature scheme. The resulting randomized public key ρA (“key-of-the-day”) is then used to index the descriptors held by the HSDir. Clients can derive the randomized key from the `.onion` address (the long-term public key or the “unblinded” version of the blinded ephemeral key) and query for a descriptor (unusable by entities without knowledge of the randomized key).

The randomized key serves as a *private index* from which the descriptor may be queried. This means that the client is implicitly checking the link between the long-term identity public key from the `.onion` address and the randomized public key. For security, it is crucial that only the actual owner of the `.onion` address can upload a descriptor to a given index. This is achieved by letting the onion services upload a signature on the descriptor, which can be verified with the randomized key. When HSDir’s verify this signature, they ensure that the descriptor is being uploaded by the actual owner of the identity public key without knowing what the `.onion` address is.

A.2 Rate-limiting Privacy Pass

In Privacy Pass, tokens are simply evaluations over a client-chosen value using a blinding protocol (*e.g.*, an OPRF or a publicly verifiable blind RSA signature).

To prevent a dictionary attack whereby the attester masquerades as a client and requests a token for a service of their choosing, clients sign in a key-blinding manner their token requests to the issuer with a secret key pp_{bsk} they know. The attester and issuer check this signature. The issuer, in turn, blinds this request with a long-term secret of their own. The attester then unblinds the response from the user and assigns this result as the identifier (the corresponding public key). To ensure that the issuer cannot use this public key and signature to link any two token requests to the same client, and clients sign their requests with a freshly chosen blind. This lets the issuer check each request for validity without letting the attester forge requests on behalf of a client.

A.3 Anonymous Credentials

Anonymous credentials (ACs), first introduced by Chaum [Cha85] and Lysanskaya *et al.* [LRSW99], allow user authentication without compromising the user’s identity. Initial constructions [CL01, CL03, CL04] consisted of a signature (representing a credential) on a commitment to the user’s identity (such as their public key) so that users could prove knowledge of their secret key to show they own a valid credential. By doing so, a trusted issuer could give out credentials to be used anonymously, *i.e.*, verifiers would only know that a user has been issued a credential without learning anything further. Unlinkable showings for the same credential could be supported if the underlying signature scheme was randomizable.

Since their introduction, the field of ACs has flourished and rapidly expanded to consider more efficient constructions and increased functionalities (*e.g.*, [PS16, CDHK15, San20, FHS15, FHS19,

[HS21]). One of the most prominent lines of works (see [KMPQ23] for a recent survey) considers anonymous attribute-based credentials (ABCs) [CDL⁺13], allowing the user to obtain a credential for an attribute set with the ability to show a subset of them in an unlinkable fashion, as done in [FHS19]. Another related notion is that of delegatable anonymous credentials (DAC) [CDD17, BB18, BCC⁺09, MSBM23], which extend ACs to allow for a delegator to issue credentials on a *root-key-owner*'s behalf. Using a DAC scheme in this way prevents a verifier from knowing which delegatee issued the credential (only that the delegatee's public key was signed by the root key). However, relying on a root authority introduces a single point of failure that can be prohibiting in some scenarios. To address this issue, very recent work studied the notion of issuer-hiding ACs (IHAC) [BEK⁺21, CLPK22, CDLPK23, MBG⁺23b] with [CLPK22] and [CDLPK23] using mercurial signatures to hide the identity of credential issuers.

Constructing DAC schemes requires additional properties beyond unlinkability and basic unforgeability, such as ensuring that the signature scheme can sign public keys and that signatures are unforgeable w.r.t. equivalence classes. This makes simple use of some signature schemes like [PS16, Gha16] for DAC not work as [PS16] has public keys in $(\mathbb{G}_2)^2$ with messages in $\mathbb{Z}_p$. Similarly, [Gha16] has public keys in $(\mathbb{G}_2)^2$ and messages in $\mathbb{G}_1 \times \mathbb{G}_2$. Furthermore, neither are unforgeable w.r.t. equivalence classes. In the mercurial signature construction from [CL19], the message and public key spaces mirror each other $((\mathbb{G}_1)^\ell$ and $(\mathbb{G}_2)^\ell$). This is why mirrored schemes can be used to construct DAC.

A.4 Deterministic wallets

Deterministic wallets can also be used in a scenario with hot and cold wallets. Here, each the wallet has its own secret, but the hot wallet is in some precarious location (typically connected to the Internet), representing a risk when a big amount of money is stored. In contrast, a cold wallet is stored offline (*e.g.*, in a hardware device). Deterministic wallets facilitate transfers between hot and cold wallets. In brief, the hot wallet randomizes the cold wallet's public key so that latter can retrieve the corresponding secret key, while keeping no secrets in the hot wallet that could lead to a forgery or privacy violation for the cold wallet.

B Classification of signature constructions

In this section, we give a brief description of how certain signature schemes approach the properties described in Sec. 4.

Pointcheval-Sanders (PS) signatures [PS16]. Derler and Slamanig present a variant of PS signatures with publicly randomizable keys [DS19]. By exponentiating the public keys with two blinding factors (one for each element in the public key) one can obtain a uniform random public key. Using those same blinding factors, one can randomize the signature to verify with the original message and the updated public key. Signatures randomized this way are identical to a fresh signature issued from the updated key on the original message, thus it achieves unlinkability with $\beta = 0$. Randomizing the keys in this way though opens up a possibility of forgery, thus making this variant only achieve 0-UNF. The attack on 1-UNF for this variant from [DS19] works as follows: The adversary queries a signature, σ , on message, m . They then choose an arbitrary message, m' . They then choose $\rho_1 = 1, \rho_2 = m/m'$ as the key randomizer and compute randomized key: $\text{pk}' = (\hat{X}', \hat{Y}') = \hat{X}\hat{Y}^{\rho_1}$ where the original key was $\text{pk} = \hat{X}\hat{Y}$. We can see that the original signature for m verifies for m' with the new key:

$$\begin{aligned} e(\sigma_1, \hat{X}'(\hat{Y}')^{m'}) &= e(\sigma_2, g_2) \\ &= e(h, g_2^x (g_2^{ym/m'})^{m'}) = e(h^{x+ym}, g_2) \\ &= e(h, g_2^x g_2^{ym}) = e(h^x h^{ym}, g_2) \end{aligned}$$

If we restrict the randomizations of PS signatures, ensuring that both elements of the public key are randomized with the same factor, we can achieve 1-UNF. Unfortunately, the public keys randomized in this way are recognizable since the owner of the secret key can compute $\hat{X}^y = \hat{Y}^x$ which holds for

any randomization of the public key. Hence, it can only achieve (1-0-3)-UNL. We present this variant below:

PS Signatures with 1-UNF and (1,0,3)-UNL

PPGen(1^λ) : Generate bilinear pairing groups, $\mathbf{p} = \mathbb{G}_1, g_1, \mathbb{G}_2, g_2, \mathbb{G}_t, e$ of prime order p .

KGen($\mathbf{p}$) : Choose $\mathbf{sk} = (x, y) \leftarrow \mathbb{Z}_p$, $\mathbf{pk} = (\hat{X}, \hat{Y}) = (g_2^x, g_2^y)$. Output $\mathbf{sk}, \mathbf{pk}$.

Sign($\mathbf{p}, \mathbf{sk}, m \in \mathbb{Z}_p$) : $h \leftarrow \mathbb{G}_1, \sigma = (\sigma_1, \sigma_2) = (h, h^{x+ym})$.

Verify($\mathbf{p}, \mathbf{pk}, m$) : Check that $e(\sigma_1, \hat{X}\hat{Y}^m) = e(\sigma_2, g_2)$.

Adapt($\mathbf{p}, \mathbf{pk}, \sigma, m, \rho$) : $\hat{X}' = \hat{X}^\rho, \hat{Y}' = \hat{Y}^\rho, r \leftarrow \mathbb{Z}_p, \sigma' = (\sigma_1^r, \sigma_2^{(\rho+ym)r})$.

We also observe that this scheme can *only sign messages in $\mathbb{Z}_p$* and so it cannot be used with GS proofs [GS08] (see Ghadafi signatures below).

Ghadafi signatures [Gha16]. Ghadafi signatures (GSig) are similar to PS signatures in that they use a randomly sampled group element to ensure unforgeability, but, GSig can sign group elements in a bilinear pairing, and in particular they are structure-preserving signatures, making them useful for GS proofs [GS08]. Note also that the signatures can be randomized while looking identical to fresh signatures, so, like PS signatures, Ghadafi signatures also achieve (1-1-3)-UNL with 0-UNF (using the variant in [DS19]). Ghadafi signatures do not have an unforgeability definition which respects equivalence classes, and so they cannot be easily used to construct delegatable anonymous credentials. The key-randomizable variant of Ghadafi signatures is provided in [DS19]. Similar to PS signatures, if both elements of the public key are randomized with the same factor, we can achieve 1-UNF, but then only satisfy (1,0,3)-UNL.

AGOT signatures [AGOT14]. Similar to PS signatures and Ghadafi signatures, we can use the variant in [DS19] to achieve (1-1-3)-UNL with 0-UNF, and use a variant with the same randomness for elements in keys to achieve 1-UNF and (1,0,3)-UNL.

Updatable signatures and signatures with honestly randomizable keys [CRS⁺21, DFL19]. These signatures have a strong assumption for their unforgeability definition, that the adversary must produce a forgery only for keys which have been honestly randomized. Thus, they only achieve 0-UNF.

Guillou-Quisquater [GQ90]. Guillou-Quisquater signatures requires a trusted setup in which an RSA modulus is generated and the secret factorization is then discarded. Because of this trusted setup, the scheme can only achieve (0-*-*)-UNL. We find a key-randomizable version of the scheme in [DS19]. We can see in this version, that a randomized public key is uniformly distributed across the set of possible public keys (since a secret key is simply an element in $\mathbb{Z}_N^*$ and randomizing it involves simply multiplying it with another secret key in $\mathbb{Z}_N^*$). Because of this (along with the fact that updated signatures look exactly like fresh signatures), the randomizable variant achieves (*-1-3)-UNL. Note also that the first element in the signature is not randomized. Thus, the scheme only achieves adaptability (Def. 4) instead of perfect adaption (Def. 5). Because one of the elements of the signature is a hash of the message, it achieves 1-UNF.

Signatures with unforgeability over equivalence classes. The signature schemes from [BHKS18, CL19, CLPK22] all achieve unforgeability with respect to message equivalence classes. With equivalence classes, one representation of a message class could be a vector of group elements. The class is then the set of messages in the message space that share some property with that representative. For example, in [CL19], the equivalence class is $\mathcal{R}_M = \{(M, M') : \exists \mu, M^\mu = M'\}$ where exponentiation is vector exponentiation and messages are $M \in \mathbb{G}_1^\ell$ for some $\ell > 1$. We define a class of a representative by $[M]_{\mathcal{R}}$ which is a set holding all message representations with the same class. We show this definition in Def. 18.

Definition 18 (α -Unforgeability w.r.t. equivalence classes). Let $\alpha \in \{0, 1\}$. A SWRK scheme is α -unforgeable if the advantage of any PPT adversary $\mathcal{A}$ defined by $\text{Adv}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UnfEquiv}}(\lambda) := \Pr[\text{Exp}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UnfEquiv}}(\lambda) \Rightarrow \text{true}] \leq \epsilon(\lambda)$, where $\text{Exp}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UnfEquiv}}(\lambda)$ is shown in Fig. 13.

Experiment $\mathbf{Exp}_{\text{SWRK}, \mathcal{A}}^{\alpha\text{-UnfEquiv}}(\lambda)$

$\Sigma_1 \leftarrow \emptyset; \Sigma_2 \leftarrow \emptyset; \text{pp} \leftarrow \text{PPGen}(1^\lambda); (\text{sk}, \text{pk}) \leftarrow \text{KGen}(\text{pp})$
 $(m^*, \sigma^*, \rho^*) \leftarrow \mathcal{A}^{\text{Sign}(\text{sk}, \cdot, \cdot), \text{Rand}(\cdot)}(\text{pk}); \text{pk}^* \leftarrow \text{RandPK}(\text{pk}, \rho^*)$
return $([m^*]_{\mathcal{R}}) \notin \Sigma_1 \wedge (\rho^* \in \Sigma_2 \vee \alpha = 1) \wedge \text{Verify}(m^*, \sigma^*, \text{pk}^*)$

Oracle $\text{Sign}(\text{sk}, m, \rho)$ Oracle $\text{Rand}()$

if $\rho \notin \Sigma_2 \wedge \alpha = 0$ **return** $\perp$ $\rho \leftarrow \mathcal{KR}$
if $\rho = \perp$ $\sigma \leftarrow \text{Sign}(\text{sk}, m)$ $\Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\}$
else $\sigma \leftarrow \text{Sign}(\text{RandSK}(\text{sk}, \rho), m)$ **return** ρ
 $\Sigma_1 \leftarrow \Sigma_1 \cup \{([m]_{\mathcal{R}})\};$ **return** σ

Fig. 13. An α -unforgeability experiment for equivalence classes.

Experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{us-euf-cma}}(\lambda, q)$

$(\text{sk}_1, \text{pk}_1) \leftarrow \text{KGen}(1^\lambda); S, T, K \leftarrow \emptyset^3, I = (\text{pk}_1, \text{sk}_1, \perp), e := 1$
 $(e^*, \sigma^*, m^*) \leftarrow \mathcal{A}^{\text{Sign}, \text{Next}, \text{Update}, \text{CorruptKey}, \text{CorruptToken}}(\text{pk}_1)$
 $S^* = \{(e', m) : e' \in K \vee (e', m) \in S \vee (e' \in T \wedge ((e' - 1, m) \in S^* \vee (e' + 1, m) \in S^*))\}$
return $\text{Verify}(\text{pk}_{e^*}, m^*, \sigma^*) \wedge \{\{(e^*, \top)\} \cup \{(e^*, m^*)\}\} \cap S^* = \emptyset$

Oracle $\text{Sign}(e', m)$ Oracle $\text{Next}(1^\lambda)$

if $e' > e$: **return** $\perp$ $(\text{pk}_{e+1}, \text{sk}_{e+1}, \rho_{e+1}) = \text{Next}(\text{pk}_e, \text{sk}_e)$
 $S = S \cup (e', m);$ $I = I \cup (\text{pk}_{e+1}, \text{sk}_{e+1}, \rho_{e+1}); e := e + 1$
return $\text{Sign}(\text{sk}_{e'}, m)$ **return** pk_e

Oracle $\text{Update}(e', m, \sigma)$ Oracle $\text{CorruptKey}(e')$ Oracle $\text{CorruptToken}(e')$

if $e' > e$: **return** $\perp$ **if** $e' > e$: **return** $\perp$ **if** $e' > e$: **return** $\perp$
if $\text{Verify}(\text{pk}_{e'}, m, \sigma) \neq 1$ **return** $\perp$ $K = K \cup \{e'\}$ $T = T \cup \{e'\}$
 $S = S \cup (e' + 1, m)$ **return** $\text{sk}_{e'}$ **return** $\rho_{e'}$
return $\text{Update}(\rho_{e'+1}, m, \sigma)$

Fig. 14. Unforgeability experiment from [CRS⁺21].

C Updatable Signature Definitions

We present the unforgeability and unlinkability definitions from [CRS⁺21] with certain aspects removed to focus on signatures (instead of signatures and MACs) and also combine the definition of a “valid” adversary (a separate definition in [CRS⁺21]) into one definition.

The unforgeability definition (Def. 19) assumes the adversary can trivially produce a signature if they corrupt either a signature from the previous key along with an update token $((e - 1, m) \in S^* \wedge e \in T)$ or a signature from the next key along with an update token $((e + 1, m) \in S^* \wedge e \in T)$. But ensures an adversary cannot produce a forgery if they do not have the appropriate keys and update tokens. The unlinkability definition (Def. ??) challenges an adversary to distinguish between a new signature or an updated one, sp the same epoch, on the same message.

Definition 19 (US-EUF-CMA [CRS⁺21]). *An updatable signature scheme Γ has existential unforgeability under chosen-message attacks for updatable signatures if, for every PPT adversary $\mathcal{A}$, the advantage function defined by $\text{Adv}_{\Gamma, \mathcal{A}}^{\text{us-euf-cma}}(\lambda) := \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{us-euf-cma}}(\lambda, q)] \leq \epsilon(\lambda)$, where the experiment $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{us-euf-cma}}(\lambda, q)$ is shown in Fig. 14⁶*

Definition 20 (Unlinkable updates under chosen-message attacks (US-UU-CMA) [CRS⁺21]). *A signature scheme, Γ , has unlinkable updates under chosen-message attacks, if for every PPT adversary $\mathcal{A}$, the advantage function defined by $\text{Adv}_{\Gamma, \mathcal{A}}^{\text{us-uu-cma}}(\lambda) := 2 \cdot \Pr[\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{us-uu-cma}}(\lambda, q) - 1] = \epsilon(\lambda)$, where $\mathbf{Exp}_{\Gamma, \mathcal{A}}^{\text{us-uu-cma}}(\lambda, q)$ is shown in Fig. 15.*

⁶ Since our focus here is on signatures, we do not consider the updatable MACs from [CRS⁺21], i.e., we can remove the verification oracle (Ver').

Experiment $\mathbf{Exp}_{\mathcal{F}, \mathcal{A}}^{\text{us-uu-cma}}(\lambda, q)$

$(\text{sk}_1, \text{pk}_1) \leftarrow \text{Setup}(1^\lambda); S \leftarrow \emptyset; (e^*, m^*) \leftarrow \mathcal{A}^{\text{Sign, Next, Update, CorruptKey, CorruptToken}}(\text{pk}_1)$
if $(\cdot, m^*) \notin S$ **return** 0; $e' = \max(\{e : (e, m^*, \cdot) \in S\})$
 $\sigma_e \leftarrow_{\$} \{\sigma : (e', m^*, \sigma) \in S\}; \forall i \in [e^* - e'] \sigma_{e'+i} = \text{Update}(\rho_{e'+i}, m^*, \sigma_{e'+i-1})$
 $\sigma^{(0)} = \sigma_{e^*}; \sigma^{(1)} = \text{Sign}(\text{sk}_{e^*}, m^*)$
 $b \leftarrow_{\$} \{0, 1\}; b' \leftarrow \mathcal{A}(\sigma^{(b)}); \text{return } e' < e^* \wedge b = b'$

Fig. 15. Unlinkable updates under chosen message attack game from [CRS⁺21].

Definition 20 uses the same oracles as in Def. 19 except that T and K are not used and S includes signatures.