SoK: Signatures With Randomizable Keys

Sofía Celi[1], Scott Griffy[2], Lucjan Hanzlik[3], Octavio Perez Kempner[4], and Daniel Slamanig[5]

[1] Brave Software, sceli@brave.com
[2] Brown University, Providence, USA, scott_griffy@brown.edu
[3] CISPA Helmholtz 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 [37] (*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.* [36], 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* [79], and subsequent formalizations by Ahn *et al.* [4] and Attrapadung *et al.* [6]. They can be considered a generalization of existing primitives, such as quotable [4, 70] or redactable signatures [98, 79, 25], homomorphic signatures for restricted classes [19, 20, 34] or any classes of functions [65] (cf. [49] 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. [36] for a more detailed discussion). In this work, we only consider the unary case. One particular class of such signatures is *randomizable signatures* [32, 33, 93]. 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.

To the best of our knowledge, the first work to explicitly study malleability on the key space is the work by Derler and Slamanig [50, 51] on *key-homomorphic* signatures inspired by previous works on pseudo-random functions [22] and encryption [21]. 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 [58] and subsequently used in [46, 5] and [55]. Kiltz *et al.* [84] 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.* [54] and [53]. Backes *et al.* [7] 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 [42] and [41]. Besides, similar notions have also been studied in works targeting related-key attacks (*e.g.,* [10, 12, 91]).

Ferreira and Dahab [57, 56] 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 [54], it considers a different case where the adversary receives a blinded secret key and must recover the long-term secret key. Furthermore, [57] and [56] 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 resembles).

There is also recent work on key-updatable signatures [78] and key-updating signatures [80] which might seem related. 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 [11]. Their main focus is on unforgeability under certain key leakages and are used in the construction of strongly secure messaging protocols. However, they are neither 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 [52]. 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 in detail, 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.

Organization. We give the preliminaries in Sec. 2. The related literature is discussed in Sec. 3. Our systematization is presented in Sec. 4. Finally, we discuss relevant applications in Sec. 5 and conclude in Sec. 6.

2 Preliminaries

To introduce the different terminology and related concepts of signatures with randomizable keys, we follow the approach of [51]. However, unlike [51], 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.

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 $\mathcal{SIG}$ (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:*

$\underline{\mathsf{PPGen}(1^\lambda)}$ *is a* PPT *algorithm that, given* λ*, outputs public parameters* pp*.*
$\underline{\mathsf{KGen}(\mathsf{pp})}$ *is a* PPT *algorithm that, given* pp*, outputs a key pair* $(\mathsf{sk}, \mathsf{pk})$*.*
$\underline{\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, m)}$ *is a* PPT *algorithm that, given* pp*, a message* m *and a secret key* sk*, outputs a signature* σ *on* m*.*
$\underline{\mathsf{Verify}(\mathsf{pp}, m, \sigma, \mathsf{pk})}$ *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*.*
$\underline{\mathsf{RandPK}(\mathsf{pp}, \mathsf{pk}, \rho)}$ *is a* PPT *algorithm that, given* pp*, a key randomizer* ρ*, and* pk*, outputs a new public key* pk' *s.t.* $\mathsf{pk}' = T(\mathsf{pk}, \rho)$ *for some transformation* T*.*
$\underline{\mathsf{RandSK}(\mathsf{pp}, \mathsf{sk}, \rho)}$ *is a* PPT *algorithm that, given* pp*,* ρ*, and* sk*, outputs a new secret key* sk' *s.t.* $\mathsf{sk}' = \widetilde{T}(\mathsf{sk}, \rho)$ *for some transformation* $\widetilde{T}$*.*
$\underline{\mathsf{Adapt}(\mathsf{pp}, [m], \sigma, \rho, \mathsf{pk})}$ *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.* $\mathsf{pk}' = T(\mathsf{pk}, \rho)$ *for some transformation* T *and outputs* (pk', σ')*.*
$\underline{\mathsf{VerKey}(\mathsf{pp}, \mathsf{sk}, \mathsf{pk})}$ *is a deterministic algorithm that takes as input* pp *and a key pair* $(\mathsf{sk}, \mathsf{pk})$*. If* $(\mathsf{sk}, \mathsf{pk})$ *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.

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

Fig. 1. Signature adaption experiment.

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

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,*

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

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 mesage-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* $\boldsymbol{Adv}^{\mathsf{Adapt}}_{\Gamma,\mathcal{A}}(\lambda) := 2 \cdot Pr\left[\boldsymbol{Exp}^{\mathsf{Adapt}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] - 1 = \epsilon(\lambda)$*, where* $\boldsymbol{Exp}^{\mathsf{Adapt}}_{\Gamma,\mathcal{A}}(\lambda)$ *is shown in Fig. 1.*

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* $\{\sigma, \mathsf{Adapt}(m, \sigma, \rho, \mathsf{pk})\}$ *and* $\{\sigma, \mathsf{RandPK}(\mathsf{pk}, \rho), \mathsf{Sign}(\mathsf{RandSK}\ (\mathsf{sk}, \rho), m)\}$ *are identical distributions where* $\mathsf{pp} \leftarrow\$ \mathsf{PPGen}(1^\lambda)$, $b \leftarrow\$ \{0,1\}$,$(\mathsf{sk},\mathsf{pk}) \leftarrow\$ \mathsf{KGen}(\mathsf{pp})$, *and* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$.

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 [62, 83, 41].

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) [71, 62] 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* [42], 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) [42] (based on [62]) in the generic group model (GGM), and ii) [41] (based on [83]) in the standard model. The second line of work is *signatures with flexible public keys* (SFPK [7]), 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 [7], in the common reference string model (CRS, [7]) and in the random oracle model (ROM, [72]). 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 [71] 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* [61, 62, 83] 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.

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 systemzatization does not consider message space randomizations. The ConvertSig algorithm in [42] 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 converter* is supplied to both functions (ConvertSig and RandPK). We can consider our Adapt function from Sec. 2 as running the two functions simultaneously. In [43], 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 [43], it will operate exactly

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

Oracle $\mathsf{Sign}(\mathsf{sk}, m)$
    return $\mathsf{Sign}(\mathsf{sk}, m)$
```

Fig. 2. Public key class-hiding experiment from [42].

like the ConvertSig from [42]. Adaptability for mercurial signatures is formalized as *origin-hiding for* ConvertSig (Def. 6) and *origin-hiding for* ChangeRep in [42].

Definition 6 (Origin-Hiding for ConvertSig [42]). *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,* $\mathsf{ConvertSig}(\sigma, \mathsf{pk}, \rho)$ *outputs a new signature σ' such that σ' is a uniform random signature in $\mathcal{SIG}$ for m, i.e.,* $\sigma' \in \{\sigma^* |\ \mathsf{Verify}(\mathsf{RandPK}(\mathsf{pk}, \rho), m, \sigma^*) = 1\}$.

We also observe that the notion of origin-hiding was formalized in a slightly different way in [41] where the authors follow the terminology from [62, 83]. 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 [42], the construction from [41] 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, [42] 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 [42]). *A mercurial signature scheme has public key class-hiding if the advantage of any* PPT *adversary $\mathcal{A}$ defined by* $\boldsymbol{Adv}^{\mathsf{PKCH\text{-}MS}}_{\Gamma,\mathcal{A}}(\lambda) := 2 \cdot Pr\left[\boldsymbol{Exp}^{\mathsf{PKCH\text{-}MS}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] - 1 = \epsilon(\lambda)$*, where* $\boldsymbol{Exp}^{\mathsf{PKCH\text{-}MS}}_{\Gamma,\mathcal{A}}(\lambda)$ *is shown in Fig. 2.*

Definition 7 is similar to *class-hiding* from [62] 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.* [7] consider equivalence classes solely on the

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

Oracle $\mathsf{Sign}(\mathsf{sk}, m)$
    $\mathbf{return}\ \mathsf{Sign}(\mathsf{sk}, m)$
```

Fig. 3. Class-hiding experiment from [7].

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 [71], which allowed them to build short static group signatures, and the first sublinear ring signature scheme without ROM or a CRS.

The *class-hiding* notion introduced in [7] 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 [13]). 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.,* [66], 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 [7] (*i.e.,* without access to the random coins).

Definition 8 (Class-Hiding [7]). *A* SFPK *scheme, Γ, has class-hiding if the advantage of any* PPT *adversary $\mathcal{A}$ in the $\boldsymbol{Exp}^{\mathsf{PKCH\text{-}SFPK}}_{\Gamma,\mathcal{A}}$ experiment is negligible as defined by $\boldsymbol{Adv}^{\mathsf{PKCH\text{-}SFPK}}_{\Gamma,\mathcal{A}}(\lambda) := 2 \cdot Pr\left[\boldsymbol{Exp}^{\mathsf{PKCH\text{-}SFPK}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{PKCH\text{-}SFPK}}_{\Gamma,\mathcal{A}}(\lambda)$ is shown in Fig. 3.*

SFPK can also be built in a way that class-hiding is conditional. In this line, Backes *et al.* [7] 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.* [7] 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 [71]. 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.* [8] 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 [8] consists of a vector of group elements, where the first element is a specified group generator.

Lastly, Hanzlik and Slamanig [72] used the same combination of SFPK and SPS-EQ (as in [7] and [8]) 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.

3.2 Signatures with Re-Randomizable Keys

Fleischhacker *et al.* [58] 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 [18] or Camenisch and Lysyanskaya [33] 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 [96], also discussed in [51] and [24]. The second one is secure in the standard model and is a variant of a scheme given by Hofheinz-Kiltz [74, 75]. Bowe *et al.* [24] refer to the Schnorr variant as a *randomizable signature* and provide a similar formalization for the unforgeability

```
Experiment $\mathbf{Exp}^{\mathsf{UNF}}_{\Gamma,\mathcal{A}}(\lambda)$
$\Sigma \leftarrow \emptyset; \mathsf{pp} \leftarrow\!\!\$\ \mathsf{PPGen}(1^\lambda); (\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\ \mathsf{KGen}(\mathsf{pp}); (\rho^*, m^*, \sigma^*) \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{pk})$
$\mathsf{pk}^* \leftarrow \mathsf{RandPK}(\mathsf{pk}, \rho^*); \mathbf{return}\ m \notin \Sigma \wedge (\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}) \vee \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}^*))$

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

Fig. 4. Unforgeability experiment from [58].

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

Oracle $\mathsf{Sign}(\mathsf{sk}, m)$
$\mathbf{return}\ \mathsf{Sign}(\mathsf{sk}, m)$
```

Fig. 5. Unlinkability experiment from [24].

property under the name of *existential unforgeability under randomization*. The difference is that the adversary in [24] can only obtain signatures for the initial secret key and not from re-randomized ones. Therefore, the unforgeability notion presented in [24] is strictly weaker than the one from [58].

Definition 9 (Unforgeability under Re-Randomized Keys [58]). *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 $\boldsymbol{Adv}^{\mathsf{UNF}}_{\Gamma,\mathcal{A}}(\lambda) := Pr\big[\boldsymbol{Exp}^{\mathsf{UNF}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] \leq \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNF}}_{\Gamma,\mathcal{A}}(\lambda)$ is shown in Fig 4.*

The authors of [58] aim to construct so called unlinkable sanitizable signatures [26, 27] and thus require unlikability 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.* [24] introduce an explicit unlinkability notion that they consider a "computational relaxation" of the previous one from [58]. Their definition (Def. 10) is close to the *class-hiding* notion from [42] 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 [24]). *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 $\boldsymbol{Adv}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) := 2\cdot\ Pr\big[\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda)$ is shown in Fig. 5.*

```
Experiment $\mathbf{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda)$
$\Sigma \leftarrow \emptyset; b \leftarrow\!\!\$ \{0,1\}; \mathsf{pp} \leftarrow\!\!\$ \mathsf{PPGen}(1^\lambda); \{\mathsf{sk}_i, \mathsf{pk}_i\}_{i\in\{0,1\}} \leftarrow\!\!\$ \mathsf{KGen}(\mathsf{pp})$
$(\rho^*, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathsf{RandPK}_1(\cdot),\mathsf{Adapt}_1(\cdot,\cdot)}(1^\lambda); \mathsf{pk}^* \leftarrow \mathsf{RandPK}(\mathsf{pk}_b, \rho^*); \Sigma \leftarrow \Sigma \cup \{(\mathsf{pk}^*, \rho^*)\}$
$b' \leftarrow\!\!\$ \mathcal{A}_2^{\mathsf{RandPK}_2(\cdot),\mathsf{Adapt}_2(\cdot,\cdot)}(\mathsf{pk}^*);$ return $b = b'$

Oracle $\mathsf{RandPK}_1(\rho)$
$\mathsf{pk}' \leftarrow \mathsf{RandPK}(\mathsf{pk}_1, \rho)$
$\Sigma \leftarrow \Sigma \cup \{(\mathsf{pk}', \rho)\}$
return $\mathsf{pk}'$

Oracle $\mathsf{RandPK}_2(\rho)$
if $\rho = \rho^*$ then return $\mathsf{pk}^*$
else $\mathsf{pk}' \leftarrow \mathsf{RandPK}(\mathsf{pk}_1, \rho); \Sigma \leftarrow \Sigma \cup \{(\mathsf{pk}', \rho)\}$
return $\mathsf{pk}'$

Oracle $\mathsf{Adapt}_1(m, \rho)$
if $\rho \notin \Sigma$ return $\perp$ else $(\sigma, \cdot) \leftarrow \mathsf{Adapt}(m, \mathsf{Sign}(\mathsf{sk}_1, m), \rho, \mathsf{pk}_1);$ return $\sigma$

Oracle $\mathsf{Adapt}_2(m, \rho)$
if $\rho \notin \Sigma$ return $\perp$; if $\rho = \rho^*$ then $(\sigma, \cdot) \leftarrow \mathsf{Adapt}(m, \mathsf{Sign}(\mathsf{sk}_b, m), \rho, \mathsf{pk}_b);$ return $\sigma$
else $(\sigma, \cdot) \leftarrow \mathsf{Adapt}(m, \mathsf{Sign}(\mathsf{sk}_1, m), \rho, \mathsf{pk}_1;$ return $\sigma$
```

Fig. 6. Unlinkability experiment from [54].

3.3 Signatures with Key Blinding

Eaton *et al.* [54] 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.

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 [54]). *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 $\boldsymbol{Adv}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) := 2\cdot Pr\big[\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda)$ is shown in Fig. 6.*

More recently, Eaton *et al.* [53] extended the study of signatures with key-blinding in an attempt to capture other applications ([54] only considered anonymity

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

Fig. 7. Unlinkability experiment from [53].

networks). The unlinkability property from [53] 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 [54], 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 [54] where no unblinding is supported.

Definition 12 (Unlinkability [53]). *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 $\boldsymbol{Adv}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) := 2\cdot Pr\big[\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNL}}_{\Gamma,\mathcal{A}}(\lambda)$ 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 [54] 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,

```
Experiment $\mathbf{Exp}^{\mathsf{SUNF}}_{\Gamma,\mathcal{A}}(\lambda)$
$\Sigma \leftarrow \emptyset$; $\mathsf{pp} \leftarrow\!\!\$\ \mathsf{PPGen}(1^\lambda)$; $(\mathsf{sk},\mathsf{pk}) \leftarrow\!\!\$\ \mathsf{KGen}(\mathsf{pp})$; $(\rho^*, m^*, \sigma^*) \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{pk})$
if $\rho^* = \perp$ then $\mathsf{pk}^* \leftarrow \mathsf{pk}$ else $\mathsf{pk}^* \leftarrow \mathsf{RandPK}(\mathsf{pk}, \rho^*)$
return $\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}^*) \wedge (\rho^*, m^*, \sigma^*) \notin \Sigma$
Oracle $\mathsf{Sign}(\mathsf{sk}, m, \rho)$
$\Sigma \leftarrow \Sigma \cup \{(\rho, m, \sigma)\}$; if $\rho = \perp$ return $\mathsf{Sign}(\mathsf{sk}, m)$; return $\mathsf{Sign}(\mathsf{RandSK}(\mathsf{sk}, \rho), m)$
```

Fig. 8. Unforgeability experiment from [53].

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 [53] also introduces a notion of *strong unforgeability*, as seen in Def. 13. This property considers any tuple of the form (ρ^*, m^*, σ^*) for which σ^* was not the result of a call to $\mathsf{Sign}(sk, m^*)$ a valid forgery. The adversary should, in this case, be able to modify σ^* or ρ^* (or both). 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 [53]). *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 $\boldsymbol{Adv}^{\mathsf{SUNF}}_{\Gamma,\mathcal{A}}(\lambda) := Pr\big[\boldsymbol{Exp}^{\mathsf{SUNF}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] \leq \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{SUNF}}_{\Gamma,\mathcal{A}}(\lambda)$ 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 [46]. For example, they require a perfect randomization property equivalent to that of [42]. 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 [46]). *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 $\boldsymbol{Adv}^{\mathsf{UNF-hrk}}_{\Gamma,\mathcal{A}}(\lambda) := Pr\big[\boldsymbol{Exp}^{\mathsf{UNF-hrk}}_{\Gamma,\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\big] \leq \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNF-hrk}}_{\Gamma,\mathcal{A}}(\lambda)$ is shown in Fig. 9.*

Def. 14 is strictly weaker than Def. 9 from Sec. 3.2, since the adversary doesn't 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.

```
Experiment $\mathbf{Exp}^{\mathsf{UNF-hrk}}_{\Gamma,\mathcal{A}}(\lambda)$
$\Sigma_1 \leftarrow \emptyset; \Sigma_2 \leftarrow \emptyset; (\mathsf{sk}, \mathsf{pk}) \leftarrow_\$ \mathsf{KGen}(1^\lambda); (m^*, \sigma^*, \rho^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot,\cdot),\mathsf{Rand}()}(\mathsf{pk})$
$\mathsf{pk}^* \leftarrow \mathsf{RandPK}(\mathsf{pk}, \rho^*);$ return $m^* \notin \Sigma_1 \wedge \rho^* \in \Sigma_2 \wedge \mathsf{Verify}(\mathsf{pk}^*, \sigma^*, m^*)$

Oracle $\mathsf{Sign}(\mathsf{sk}, m, \rho)$
If $\rho \notin \Sigma_2$, return $\perp; \mathsf{sk}' \leftarrow \mathsf{RandSK}(\mathsf{sk}, \rho)$
$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}', m); \Sigma_1 \leftarrow \Sigma_1 \cup \{m\};$ return $\sigma$

Oracle $\mathsf{Rand}()$
$\rho \leftarrow_\$ \mathcal{KR}; \Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\};$ return $\rho$
```

Fig. 9. Unforgeability under honestly rerandomized keys experiment from [46].

3.5 Updatable signatures

Updatable signatures [39] are the signature equivalent to updatable encryption (UE) [22], 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.* [39] consider constructions from key-homomorphic signatures [51] 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 Appendix C (Def. 19). In the game, the adversary can use the oracles to obtain signatures, updates of signatures (Update), public keys (Next), secret keys ($\mathsf{CorruptKey}$), and update tokens ($\mathsf{CorruptToken}$), with the restriction that trivial forgeries are excluded. To capture them, the authors in analogy to UE in [87] create a recursively defined set (S^* in Def. 19) of messages and epochs that the adversary must be able to compute by correctness (cf. [39] 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.* [39] 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 Appendix C. 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).

Klooß *et al.* [86] construct a one-time signature (instantiated based on the one-time SPS from [85]) 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 [85]) remains open.

```
Experiment $\mathbf{Exp}^{\alpha-\mathsf{UNF}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$
$\Sigma_1 \leftarrow \emptyset; \Sigma_2 \leftarrow \emptyset; \mathsf{pp} \leftarrow\!\!\$\ \mathsf{PPGen}(1^\lambda); (\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\ \mathsf{KGen}(\mathsf{pp})$
$(m^*, \sigma^*, \rho^*) \leftarrow\!\!\$\ \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot,\cdot),\mathsf{Rand}()}(\mathsf{pk}); \mathsf{pk}^* \leftarrow \mathsf{RandPK}(\mathsf{pk}, \rho^*)$
$\mathbf{return}\ (m^* \boxed{, \sigma^*}) \notin \Sigma_1 \wedge (\rho^* \in \Sigma_2 \vee \alpha = 1) \wedge \mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}^*)$
```

```
Oracle $\mathsf{Sign}(\mathsf{sk}, m, \rho)$
$\mathbf{if}\ \rho \notin \Sigma_2 \wedge \alpha = 0\ \mathbf{return}\ \bot$
$\mathbf{if}\ \rho = \bot\ \sigma \leftarrow\!\!\$\ \mathsf{Sign}(\mathsf{sk}, m)$
$\mathbf{else}\ \sigma \leftarrow \mathsf{Sign}(\mathsf{RandSK}(\mathsf{sk}, \rho), m)$
$\Sigma_1 \leftarrow \Sigma_1 \cup \{(m \boxed{, \sigma})\}; \mathbf{return}\ \sigma$
```

```
Oracle $\mathsf{Rand}()$
$\rho \leftarrow_\$ \mathcal{KR}$
$\Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\}$
$\mathbf{return}\ \rho$
```

Fig. 10. Our α-unforgeability experiment. $\boxed{\text{Solid boxes}}$ refer to *strong* unforgeability.

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 [58] and [46] (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 $(m^*, \sigma^*, \rho^*, \mathsf{pk}^*)$ to trivially win the game. As in [53], 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 $\boldsymbol{Adv}^{\alpha-\mathsf{UNF}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) := Pr\left[\boldsymbol{Exp}^{\alpha-\mathsf{UNF}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] \leq \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{UNF}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$ 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 [92]. 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

```
Experiment $\mathbf{Exp}^{(\alpha,\beta,\gamma)-\mathsf{UNL}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$
$\mathsf{st} \leftarrow \emptyset; b \leftarrow\!\!\$ \{0,1\}; \rho \leftarrow\!\!\$ \mathcal{KC};$ if $\alpha = 0$ then $\mathsf{pp} \leftarrow\!\!\$ \mathsf{PPGen}(1^\lambda)$ else $(\mathsf{pp},\mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{st}, 1^\lambda)$
if $\beta = 0$ then $\{\mathsf{sk}_i, \mathsf{pk}_i\}_{i\in\{0,1\}} \leftarrow\!\!\$ \mathsf{KGen}(\mathsf{pp})$ else $(\{\mathsf{sk}_i,\mathsf{pk}_i\}_{i\in\{0,1\}}, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{st},\mathsf{pp})$
if $\exists\ i \in \{0,1\}:\ \mathsf{VerKey}(\mathsf{sk}_i,\mathsf{pk}_i) = 0$ return $0$
$\mathsf{pk}' \leftarrow \mathsf{RandPK}(\mathsf{pk}_b, \rho); \mathsf{sk}' \leftarrow \mathsf{RandSK}(\mathsf{sk}_b, \rho)$
if $\gamma = \emptyset$ then $b' \leftarrow\!\!\$ \mathcal{A}_2(\mathsf{pk}', \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{st})$ else $b' \leftarrow\!\!\$ \mathcal{A}_2^{\mathsf{Sign}_\gamma(\cdot)}(\mathsf{pk}', \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{st})$
return $b = b'$

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

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

the signing oracle available to the adversary. For ease of exposition, we supply γ to a function K[6], resulting in the following scenarios to consider:

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

Definition 16 ((α,β,γ)-Unlinkability). *Let α, $\beta \in \{0,1\}$ and γ 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 $\boldsymbol{Adv}^{(\alpha,\beta,\gamma)-\mathsf{UNL}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) := 2 \cdot Pr\left[\boldsymbol{Exp}^{(\alpha,\beta,\gamma)-\mathsf{UNL}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{(\alpha,\beta,\gamma)-\mathsf{UNL}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$ 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 the adversary for configurations $(\cdot, 0, \cdot)$. However, as discussed in [7], 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

[6] We will use the shorthand forms of 0, 1 or 3 to instantiate γ.

```
Experiment $\mathbf{Exp}^{(\alpha,\beta)-\mathsf{UNE}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$
$\mathsf{st} \leftarrow \emptyset; \gamma \leftarrow \perp; b \leftarrow\!\!\$\ \{0,1\};$ if $\alpha = 0$ then $\mathsf{pp} \leftarrow\!\!\$\ \mathsf{PPGen}(1^\lambda)$ else $(\mathsf{pp},\mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{st}, 1^\lambda)$
if $\beta = 0$ then $\rho \leftarrow\!\!\$\ \mathcal{KC}$ else $(\rho, \mathsf{st}) \leftarrow \mathcal{A}_1(\mathsf{st}, \mathsf{pp}); \{\mathsf{sk}_i, \mathsf{pk}_i\}_{i\in\{0,1\}} \leftarrow\!\!\$\ \mathsf{KGen}(\mathsf{pp})$
$\mathsf{pk}'_b \leftarrow \mathsf{RandPK}(\mathsf{pk}_b, \rho); \mathsf{sk}'_b \leftarrow \mathsf{RandSK}(\mathsf{sk}_b, \rho); b' \leftarrow\!\!\$\ \mathcal{A}_2^{\mathsf{Sign}(\cdot,\cdot)}(\mathsf{pk}'_b, \rho, \mathsf{st});$ return $b = b'$

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

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

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 α, $\beta \in \{0, 1\}$. 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 $\boldsymbol{Adv}^{(\alpha,\beta)-\mathsf{UNE}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) := 2 \cdot Pr\Big[\boldsymbol{Exp}^{(\alpha,\beta)-\mathsf{UNE}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\Big] - 1 = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{(\alpha,\beta)-\mathsf{UNE}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$ 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 to as discussed in Appendix B.

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

Scheme	A	PA	α-UNF	(α,β,γ)-UNL	(α,β)-UNE	Setting	$\vert\mathsf{pk}\vert$	$\vert\sigma\vert$
Schnorr [97, 24, 58]	✓	✗	1-UNF	(1,0,3*)	✗	(EC)DL & ROM	$1\vert\mathbb{G}\vert$	$1\vert\mathbb{Z}_p\vert + 1\vert\mathbb{G}\vert$
BLS [23, 51]	✓	✓	1-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}I}}$ & ROM	$1\vert\mathbb{G}\vert$	$1\vert\mathbb{G}\vert$
Katz-Wang [82, 64, 51]	✓	✗	1-UNF	(1,0,3)	✗	(EC)DL & ROM	$2\vert\mathbb{G}\vert$	$2\vert\mathbb{G}\vert + 1\vert\mathbb{Z}_p\vert$
Guillou-Quisquater [69, 51]	✓	✗	1-UNF	(0,1,3)	✗	RSA(CRS) & ROM	$1\vert\mathbb{Z}_N\vert$	$1\vert\mathbb{Z}_e\vert + 1\vert\mathbb{Z}_N\vert^{**}$
Waters [100, 14, 51]	✓	✓	1-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}III}}$ & CDH	$1\vert\mathbb{G}_2\vert$	$1\vert\mathbb{G}_1\vert + 2\vert\mathbb{G}_2\vert$
Pointcheval-Sanders [93, 51]	✓	✓	0-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}III}}$ & GGM	$2\vert\mathbb{G}_2\vert$	$2\vert\mathbb{G}_1\vert$
AGOT [3, 51]	✓	✓	0-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}II}}$ & GGM	$2\vert\mathbb{G}_1\vert$	$2\vert\mathbb{G}_2\vert$
Ghadafi [63, 51]	✓	✓	0-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}III}}$ & GGM	$2\vert\mathbb{G}_2\vert$	$3\vert\mathbb{G}_1\vert$
EdDSA [53]	✗	✗	1-UNF†	(1,1,3)	$(0,1)$-UNE	(EC)DL	$1\vert\mathbb{G}\vert$	$1\vert\mathbb{G}\vert + 1\vert\mathbb{Z}_p\vert$
ECDSA [53]	✓	✗	1-UNF	(1,1,3)	$(0,1)$-UNE	(EC)DL	$1\vert\mathbb{G}\vert$	$2\vert\mathbb{Z}_p\vert$
Hofheinz-Kiltz [74, 75, 58]	✗	✗	1-UNF	(1,1,3)	✗	BG & q-SDH	$1\vert\mathbb{G}_2\vert$	$1\vert\mathbb{Z}_p\vert + 1\vert\mathbb{G}_1\vert$
BHKS1 (Scheme 4) [7]	✓	✓	1-UNF‡	(1,0,3*)	✗	BG & DLIN & DDH	$(\ell+5)\vert\mathbb{G}_1\vert$	$2\vert\mathbb{G}_1\vert + 1\vert\mathbb{G}_2\vert$
BHKS2 (Scheme 5) [7]	✓	✓	1-UNF‡	(0,0,3*)	✗	BG & DLIN & DDH	$3\vert\mathbb{G}_1\vert$	$2\vert\mathbb{G}_1\vert + 1\vert\mathbb{G}_2\vert$
Crites-Lysyanskaya [42]	✓	✓	1-UNF‡	(1,0,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}III}}$ & GGM	$\ell\vert\mathbb{G}_1\vert$	$2\vert\mathbb{G}_1\vert + 1\vert\mathbb{G}_2\vert$
CLPK [41]	✓	✓	1-UNF‡	(0,0,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}III}}$ & CRS	$(2+\ell)\vert\mathbb{G}_2\vert$	$9\vert\mathbb{G}_1\vert + 4\vert\mathbb{G}_2\vert$
ESS (Dilithium) [54]	✗	✗	1-UNF	(1,0,3)	$(0,1)$-UNE	MLWE	7.7kB	5.7kB
ESS (Picnic) [54]	✗	✗	1-UNF	(1,0,3)	$(0,1)$-UNE	MPC-in-the-head	32B	$\approx$ 75kB***
ESS (LegRoast) [54]	✗	✗	1-UNF	(1,0,3)	$(0,1)$-UNE	PRF	0.50kB	7.94kB
ESS (CSI-FiSh) [54]	✗	✗	1-UNF	(1,0,3)	$(0,1)$-UNE	CSIDH	32B	1.8-2.1kB
US (Fig. 7) [39]	✓	✓	1-UNF	(1,1,3)	✗	$\mathrm{BG}^{\mathsf{Type\text{-}I}}$ & ROM	$1\vert\mathbb{G}\vert$	$1\vert\mathbb{G}\vert$
HRK (Fig. 10) [46]	✗	✗	0-UNF	(1,1,3)	✗	(EC)DL	$1\vert\{0,1\}^{\lambda}\vert + 2\vert\mathbb{Z}_p\vert$	$1\vert\mathbb{G}\vert$

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 ((α,β,γ)-UNL, where $*$ means that keys can be leaked to the adversary) and unextractability ((α,β)-UNE) properties. We also include the setting in which they work, and the size of pk (ℓ parametrizes the message vector's length) and σ.
$^{**}e$ is a large prime roughly the size of $\phi(N)$. *** Estimated size.

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 $\mathsf{sk}' = \mathsf{sk} \cdot \mathsf{H}(\mathsf{Sign}(\mathsf{sk}, \text{“0”}), \mathsf{pk})$. 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 $\mathsf{H}(\mathsf{Sign}(\mathsf{sk}, \text{“0”}), \mathsf{pk})$. This can be easily shown 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 $\mathsf{H}(\mathsf{Sign}(\mathsf{sk}_0, \text{“0”}), \mathsf{pk}_0)$ and $\mathsf{H}(\mathsf{Sign}(\mathsf{sk}_1, \text{“0”}), \mathsf{pk}_1)$, and check which public key corresponds to 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 [99], the Ed25519 signature scheme [15] 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 $\rho \leftarrow H(\tau \parallel A)$ with $\rho \in [\ell - 1]$. The randomized key pair, $(\rho \cdot a \mod l, A^\rho)$, is entirely fresh and compatible with Ed25519 for signing and verification (see Appendix A.1 for more details). The application requires the scheme to provide two essential properties: *unlinkability* and *unforgeability*. As discussed by [54, 76], 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 [53]).

5.2 Rate-limiting Privacy Pass

Privacy Pass [47] 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 [35, 48] 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 [73] 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 (see Appendix A.2), the protocol requires unlinkability, and unforgeability. As discussed by [53], 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. [53] cites this scheme as an example of "key-blinding with unblinding".

5.3 Anonymous Credentials

Anonymous credentials (see Appendix A.3 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 [42, 43, 17, 41, 40, 89]. Potential solutions to this problem require different properties. In issuer-hiding credentials (*e.g.,* [17, 41]), 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 [42, 60]). 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 [42]. 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 also Appendix A.4 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.* [94], stealth addresses are closely related to asynchronous remote key generation for FIDO tokens [59]. 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 [51], 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 [2, 1], 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 [51] to build zk-SNARKs and circuit-succinct NIZK proofs with an updatable CRS [67].

6 Conclusions

6.1 Future work

As we saw in Sec. 5, SWRK have an array of practical, real-world privacy-preserving applications. A first direction for future work 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. Third and very important direction are post-quantum constructions: there are already some first works available [5, 54, 77, 44, 45]. 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. Finally, 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.

6.2 Final Remarks

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. As a result, we identified exciting areas to explore as previously outlined.

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

1. Abdolmaleki, B., Glaeser, N., Ramacher, S., Slamanig, D.: Universally composable NIZKs: Circuit-succinct, non-malleable and CRS-updatable. Cryptology ePrint Archive, Report 2023/097 (2023), https://eprint.iacr.org/2023/097
2. Abdolmaleki, B., Ramacher, S., Slamanig, D.: Lift-and-shift: Obtaining simulation extractable subversion and updatable SNARKs generically. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1987–2005. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3417228
3. Abe, M., Groth, J., Ohkubo, M., Tibouchi, M.: Structure-preserving signatures from type II pairings. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 390–407. Springer, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_22
4. Ahn, J.H., Boneh, D., Camenisch, J., Hohenberger, S., shelat, a., Waters, B.: Computing on authenticated data. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 1–20. Springer, Heidelberg (Mar 2012). https://doi.org/10.1007/978-3-642-28914-9_1
5. Alkeilani Alkadri, N., Das, P., Erwig, A., Faust, S., Krämer, J., Riahi, S., Struck, P.: Deterministic wallets in a quantum world. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1017–1031. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3423361
6. Attrapadung, N., Libert, B., Peters, T.: Computing on authenticated data: New privacy definitions and constructions. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 367–385. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4_23
7. Backes, M., Hanzlik, L., Kluczniak, K., Schneider, J.: Signatures with flexible public key: Introducing equivalence classes for public keys. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part II. LNCS, vol. 11273, pp. 405–434. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03329-3_14

8. Backes, M., Hanzlik, L., Schneider-Bensch, J.: Membership privacy for fully dynamic group signatures. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 2181–2198. ACM Press (Nov 2019). https://doi.org/10.1145/3319535.3354257
9. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (Aug 2009). https://doi.org/10.1007/978-3-642-03356-8_7
10. Bellare, M., Cash, D., Miller, R.: Cryptography secure against related-key attacks and tampering. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 486–503. Springer, Heidelberg (Dec 2011). https://doi.org/10.1007/978-3-642-25385-0_26
11. Bellare, M., Miner, S.K.: A forward-secure digital signature scheme. In: Wiener, M.J. (ed.) CRYPTO'99. LNCS, vol. 1666, pp. 431–448. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1_28
12. Bellare, M., Paterson, K.G., Thomson, S.: RKA security beyond the linear barrier: IBE, encryption and signatures. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 331–348. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4_21
13. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. Journal of Cryptology **22**(1), 114–138 (Jan 2009). https://doi.org/10.1007/s00145-007-9011-9
14. Bernhard, D., Fuchsbauer, G., Ghadafi, E.: Efficient signatures of knowledge and DAA in the standard model. In: Jacobson Jr., M.J., Locasto, M.E., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 13. LNCS, vol. 7954, pp. 518–533. Springer, Heidelberg (Jun 2013). https://doi.org/10.1007/978-3-642-38980-1_33
15. Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y.: High-speed high-security signatures. Journal of Cryptographic Engineering **2**(2), 77–89 (Sep 2012). https://doi.org/10.1007/s13389-012-0027-1
16. Blömer, J., Bobolz, J.: Delegatable attribute-based anonymous credentials from dynamically malleable signatures. In: Preneel, B., Vercauteren, F. (eds.) ACNS 18. LNCS, vol. 10892, pp. 221–239. Springer, Heidelberg (Jul 2018). https://doi.org/10.1007/978-3-319-93387-0_12
17. Bobolz, J., Eidens, F., Krenn, S., Ramacher, S., Samelin, K.: Issuer-hiding attribute-based credentials. In: Conti, M., Stevens, M., Krenn, S. (eds.) CANS 21. LNCS, vol. 13099, pp. 158–178. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92548-2_9
18. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (May 2004). https://doi.org/10.1007/978-3-540-24676-3_4
19. Boneh, D., Freeman, D., Katz, J., Waters, B.: Signing a linear subspace: Signature schemes for network coding. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 68–87. Springer, Heidelberg (Mar 2009). https://doi.org/10.1007/978-3-642-00468-1_5
20. Boneh, D., Freeman, D.M.: Homomorphic signatures for polynomial functions. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 149–168. Springer, Heidelberg (May 2011). https://doi.org/10.1007/978-3-642-20465-4_10
21. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald,

E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_30
22. Boneh, D., Lewi, K., Montgomery, H.W., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_23
23. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. Journal of Cryptology **17**(4), 297–319 (Sep 2004). https://doi.org/10.1007/s00145-004-0314-9
24. Bowe, S., Chiesa, A., Green, M., Miers, I., Mishra, P., Wu, H.: ZEXE: Enabling decentralized private computation. In: 2020 IEEE Symposium on Security and Privacy. pp. 947–964. IEEE Computer Society Press (May 2020). https://doi.org/10.1109/SP40000.2020.00050
25. Brzuska, C., Busch, H., Dagdelen, Ö., Fischlin, M., Franz, M., Katzenbeisser, S., Manulis, M., Onete, C., Peter, A., Poettering, B., Schröder, D.: Redactable signatures for tree-structured data: Definitions and constructions. In: Zhou, J., Yung, M. (eds.) ACNS 10. LNCS, vol. 6123, pp. 87–104. Springer, Heidelberg (Jun 2010). https://doi.org/10.1007/978-3-642-13708-2_6
26. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of sanitizable signatures revisited. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 317–336. Springer, Heidelberg (Mar 2009). https://doi.org/10.1007/978-3-642-00468-1_18
27. Brzuska, C., Fischlin, M., Lehmann, A., Schröder, D.: Unlinkability of sanitizable signatures. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 444–461. Springer, Heidelberg (May 2010). https://doi.org/10.1007/978-3-642-13013-7_26
28. Camenisch, J., Drijvers, M., Dubovitskaya, M.: Practical UC-secure delegatable credentials with attributes and their application to blockchain. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) ACM CCS 2017. pp. 683–699. ACM Press (Oct / Nov 2017). https://doi.org/10.1145/3133956.3134025
29. Camenisch, J., Dubovitskaya, M., Haralambiev, K., Kohlweiss, M.: Composable and modular anonymous credentials: Definitions and practical constructions. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015, Part II. LNCS, vol. 9453, pp. 262–288. Springer, Heidelberg (Nov / Dec 2015). https://doi.org/10.1007/978-3-662-48800-3_11
30. Camenisch, J., Dubovitskaya, M., Lehmann, A., Neven, G., Paquin, C., Preiss, F.S.: Concepts and languages for privacy-preserving attribute-based authentication. In: Fischer-Hübner, S., de Leeuw, E., Mitchell, C. (eds.) Policies and Research in Identity Management. pp. 34–52. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
31. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (May 2001). https://doi.org/10.1007/3-540-44987-6_7
32. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 02. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (Sep 2003). https://doi.org/10.1007/3-540-36413-7_20
33. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (Aug 2004). https://doi.org/10.1007/978-3-540-28628-8_4

34. Catalano, D., Fiore, D., Warinschi, B.: Homomorphic signatures with efficient verification for polynomial functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 371–389. Springer, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_21
35. Celi, S., Davidson, A., Valdez, S., Wood, C.: Privacy pass issuance protocol (2023), https://datatracker.ietf.org/doc/draft-ietf-privacypass-protocol/
36. Chase, M., Kohlweiss, M., Lysyanskaya, A., Meiklejohn, S.: Malleable signatures: New definitions and delegatable anonymous credentials. In: Datta, A., Fournet, C. (eds.) CSF 2014 Computer Security Foundations Symposium. pp. 199–213. IEEE Computer Society Press (2014). https://doi.org/10.1109/CSF.2014.22
37. Chator, A., Green, M., Tiwari, P.R.: Sok: Privacy-preserving signatures. IACR Cryptol. ePrint Arch. p. 1039 (2023), https://eprint.iacr.org/2023/1039
38. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Commun. ACM **28**(10), 1030–1044 (oct 1985), https://doi.org/10.1145/4372.4373
39. Cini, V., Ramacher, S., Slamanig, D., Striecks, C., Tairi, E.: Updatable signatures and message authentication codes. In: Garay, J. (ed.) PKC 2021, Part I. LNCS, vol. 12710, pp. 691–723. Springer, Heidelberg (May 2021). https://doi.org/10.1007/978-3-030-75245-3_25
40. Connolly, A., Deschamps, J., Lafourcade, P., Perez Kempner, O.: 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. p. 249–271. Springer-Verlag, Berlin, Heidelberg (2023), https://doi.org/10.1007/978-3-031-22912-1_11
41. Connolly, A., Lafourcade, P., Perez-Kempner, O.: Improved constructions of anonymous credentials from structure-preserving signatures on equivalence classes. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) PKC 2022, Part I. LNCS, vol. 13177, pp. 409–438. Springer, Heidelberg (Mar 2022). https://doi.org/10.1007/978-3-030-97121-2_15
42. Crites, E.C., Lysyanskaya, A.: Delegatable anonymous credentials from mercurial signatures. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 535–555. Springer, Heidelberg (Mar 2019). https://doi.org/10.1007/978-3-030-12612-4_27
43. Crites, E.C., Lysyanskaya, A.: Mercurial signatures for variable-length messages. PoPETs **2021**(4), 441–463 (Oct 2021). https://doi.org/10.2478/popets-2021-0079
44. Das, P., Erwig, A., Faust, S., Loss, J., Riahi, S.: BIP32-compatible threshold wallets. Cryptology ePrint Archive, Report 2023/312 (2023), https://eprint.iacr.org/2023/312
45. Das, P., Erwig, A., Meyer, M., Struck, P.: Efficient post-quantum secure deterministic threshold wallets from isogenies. Cryptology ePrint Archive, Paper 2023/1915 (2023), https://eprint.iacr.org/2023/1915, https://eprint.iacr.org/2023/1915
46. Das, P., Faust, S., Loss, J.: A formal treatment of deterministic wallets. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 651–668. ACM Press (Nov 2019). https://doi.org/10.1145/3319535.3354236
47. Davidson, A., Goldberg, I., Sullivan, N., Tankersley, G., Valsorda, F.: Privacy pass: Bypassing internet challenges anonymously. PoPETs **2018**(3), 164–180 (Jul 2018). https://doi.org/10.1515/popets-2018-0026

48. Davidson, A., Iyengar, J., Wood, C.: The privacy pass architecture (2023), https://datatracker.ietf.org/doc/draft-ietf-privacypass-architecture/
49. Demirel, D., Derler, D., Hanser, C., Pöhls, H., Slamanig, D., Traverso, G.: Prismacloud d4.4: Overview of functional and malleable signature schemes (2015)
50. Derler, D., Slamanig, D.: Key-homomorphic signatures and applications to multiparty signatures. Cryptology ePrint Archive, Report 2016/792 (2016), https://eprint.iacr.org/2016/792
51. Derler, D., Slamanig, D.: Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. Designs, Codes and Cryptography **87**(6), 1373–1413 (Jun 2019), https://doi.org/10.1007/s10623-018-0535-9
52. Dowling, B., Hauck, E., Riepel, D., Rösler, P.: Strongly anonymous ratcheted key exchange. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part III. LNCS, vol. 13793, pp. 119–150. Springer, Heidelberg (Dec 2022). https://doi.org/10.1007/978-3-031-22969-5_5
53. Eaton, E., Lepoint, T., Wood, C.A.: Security analysis of signature schemes with key blinding. Cryptology ePrint Archive, Paper 2023/380 (2023), https://eprint.iacr.org/2023/380
54. Eaton, E., Stebila, D., Stracovsky, R.: Post-quantum key-blinding for authentication in anonymity networks. In: Longa, P., Ràfols, C. (eds.) LATINCRYPT 2021. LNCS, vol. 12912, pp. 67–87. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-88238-9_4
55. Erwig, A., Riahi, S.: Deterministic wallets for adaptor signatures. In: Atluri, V., Di Pietro, R., Jensen, C.D., Meng, W. (eds.) ESORICS 2022, Part II. LNCS, vol. 13555, pp. 487–506. Springer, Heidelberg (Sep 2022). https://doi.org/10.1007/978-3-031-17146-8_24
56. Ferreira, L., Dahab, R.: Optimistic blinded-key signatures. In: IEEE First Symposium onMulti-Agent Security and Survivability, 2004. pp. 65–72 (2004). https://doi.org/10.1109/MASSUR.2004.1368419
57. Ferreira, L.C., Dahab, R.: Blinded-key signatures: Securing private keys embedded in mobile agents. In: Proceedings of the 2002 ACM Symposium on Applied Computing. p. 82–86. SAC '02, Association for Computing Machinery, New York, NY, USA (2002), https://doi.org/10.1145/508791.508808
58. Fleischhacker, N., Krupp, J., Malavolta, G., Schneider, J., Schröder, D., Simkin, M.: Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. In: Cheng, C.M., Chung, K.M., Persiano, G., Yang, B.Y. (eds.) PKC 2016, Part I. LNCS, vol. 9614, pp. 301–330. Springer, Heidelberg (Mar 2016). https://doi.org/10.1007/978-3-662-49384-7_12
59. Frymann, N., Gardham, D., Kiefer, F., Lundberg, E., Manulis, M., Nilsson, D.: Asynchronous remote key generation: An analysis of yubico's proposal for W3C WebAuthn. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 939–954. ACM Press (Nov 2020). https://doi.org/10.1145/3372297.3417292
60. Fuchsbauer, G.: Commuting signatures and verifiable encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 224–245. Springer, Heidelberg (May 2011). https://doi.org/10.1007/978-3-642-20465-4_14
61. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 233–253. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_12

62. Fuchsbauer, G., Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. Journal of Cryptology **32**(2), 498–546 (Apr 2019). https://doi.org/10.1007/s00145-018-9281-4
63. Ghadafi, E.: Short structure-preserving signatures. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 305–321. Springer, Heidelberg (Feb / Mar 2016). https://doi.org/10.1007/978-3-319-29485-8_18
64. Goh, E.J., Jarecki, S., Katz, J., Wang, N.: Efficient signature schemes with tight reductions to the Diffie-Hellman problems. Journal of Cryptology **20**(4), 493–514 (Oct 2007). https://doi.org/10.1007/s00145-007-0549-3
65. Gorbunov, S., Vaikuntanathan, V., Wichs, D.: Leveled fully homomorphic signatures from standard lattices. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC. pp. 469–477. ACM Press (Jun 2015). https://doi.org/10.1145/2746539.2746576
66. Griffy, S., Lysyanskaya, A.: PACIFIC: Privacy-preserving automated contact tracing scheme featuring integrity against cloning. Cryptology ePrint Archive, Report 2023/371 (2023), https://eprint.iacr.org/2023/371
67. Groth, J., Kohlweiss, M., Maller, M., Meiklejohn, S., Miers, I.: Updatable and universal common reference strings with applications to zk-SNARKs. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 698–728. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96878-0_24
68. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (Apr 2008). https://doi.org/10.1007/978-3-540-78967-3_24
69. Guillou, L.C., Quisquater, J.J.: A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO'88. LNCS, vol. 403, pp. 216–231. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34799-2_16
70. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: Abe, M., Gligor, V. (eds.) ASIACCS 08. pp. 353–362. ACM Press (Mar 2008)
71. Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 491–511. Springer, Heidelberg (Dec 2014). https://doi.org/10.1007/978-3-662-45611-8_26
72. Hanzlik, L., Slamanig, D.: With a little help from my friends: Constructing practical anonymous credentials. In: Vigna, G., Shi, E. (eds.) ACM CCS 2021. pp. 2004–2023. ACM Press (Nov 2021). https://doi.org/10.1145/3460120.3484582
73. Hendrickson, S., Iyengar, J., Pauly, T., Valdez, S., Wood, C.: Rate-limited token issuance protocol (2023), https://datatracker.ietf.org/doc/draft-ietf-privacypass-rate-limit-tokens/02/
74. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (Aug 2008). https://doi.org/10.1007/978-3-540-85174-5_2
75. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. Journal of Cryptology **25**(3), 484–527 (Jul 2012). https://doi.org/10.1007/s00145-011-9102-5
76. Hopper, N.: Proving security of tor's hidden service identity blinding protocol (2013), https://www-users.cse.umn.edu/~hoppernj/basic-proof.pdf

77. Hu, M.: 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
78. Jaeger, J., Stepanovs, I.: Optimal channel security against fine-grained state compromise: The safety of messaging. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 33–62. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96884-1_2
79. Johnson, R., Molnar, D., Song, D.X., Wagner, D.: Homomorphic signature schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (Feb 2002). https://doi.org/10.1007/3-540-45760-7_17
80. Jost, D., Maurer, U., Mularczyk, M.: Efficient ratcheting: Almost-optimal guarantees for secure messaging. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part I. LNCS, vol. 11476, pp. 159–188. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17653-2_6
81. Kakvi, S.A., Martin, K.M., Putman, C., Quaglia, E.A.: Sok: Anonymous credentials. In: Günther, F., Hesse, J. (eds.) Security Standardisation Research. pp. 129–151. Springer Nature Switzerland, Cham (2023)
82. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) ACM CCS 2003. pp. 155–164. ACM Press (Oct 2003). https://doi.org/10.1145/948109.948132
83. Khalili, M., Slamanig, D., Dakhilalian, M.: Structure-preserving signatures on equivalence classes from standard assumptions. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part III. LNCS, vol. 11923, pp. 63–93. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-34618-8_3
84. Kiltz, E., Masny, D., Pan, J.: Optimal security proofs for signatures from identification schemes. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 33–61. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53008-5_2
85. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 275–295. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_14
86. Klooß, M., Lehmann, A., Rupp, A.: (R)CCA secure updatable encryption with integrity protection. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part I. LNCS, vol. 11476, pp. 68–99. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17653-2_3
87. Lehmann, A., Tackmann, B.: Updatable encryption with post-compromise security. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 685–716. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78372-7_22
88. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-46513-8_14
89. Mir, O., Bauer, B., Griffy, S., Lysyanskaya, A., Slamanig, D.: Aggregate signatures with versatile randomization and issuer-hiding multi-authority anonymous credentials. Cryptology ePrint Archive, Paper 2023/1016 (2023), https://eprint.iacr.org/2023/1016
90. Mir, O., Slamanig, D., Bauer, B., Mayrhofer, R.: Practical delegatable anonymous credentials from equivalence class signatures. Proc. Priv. Enhancing Technol. **2023**(3), 488–513 (2023), https://doi.org/10.56553/popets-2023-0093

91. Morita, H., Schuldt, J.C.N., Matsuda, T., Hanaoka, G., Iwata, T.: On the security of the schnorr signature scheme and DSA against related-key attacks. In: Kwon, S., Yun, A. (eds.) ICISC 15. LNCS, vol. 9558, pp. 20–35. Springer, Heidelberg (Nov 2016). https://doi.org/10.1007/978-3-319-30840-1_2
92. Park, S., Sealfon, A.: It wasn't me! - Repudiability and claimability of ring signatures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 159–190. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_6
93. Pointcheval, D., Sanders, O.: Short randomizable signatures. In: Sako, K. (ed.) CT-RSA 2016. LNCS, vol. 9610, pp. 111–126. Springer, Heidelberg (Feb / Mar 2016). https://doi.org/10.1007/978-3-319-29485-8_7
94. Pu, S., Thyagarajan, S.A., Döttling, N., Hanzlik, L.: Post quantum fuzzy stealth signatures and applications. Cryptology ePrint Archive, Paper 2023/1148 (2023), https://eprint.iacr.org/2023/1148
95. Sanders, O.: Efficient redactable signature and application to anonymous credentials. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part II. LNCS, vol. 12111, pp. 628–656. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45388-6_22
96. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34805-0_22
97. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology **4**(3), 161–174 (Jan 1991). https://doi.org/10.1007/BF00196725
98. Steinfeld, R., Bull, L., Zheng, Y.: Content extraction signatures. In: Kim, K. (ed.) ICISC 01. LNCS, vol. 2288, pp. 285–304. Springer, Heidelberg (Dec 2002)
99. The Tor Project, I.: Tor rendezvous specification - version 3 (2023), https://gitweb.torproject.org/torspec.git/tree/rend-spec-v3.txt
100. Waters, B.R.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_7

Appendix

A Applications: Detailed discussion

A.1 Anonymity networks

We recall the exposition from [54]. 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 [38] and Lysanskaya *et al.* [88], allow user authentication without compromising the user's identity. Initial constructions [31, 32, 33] 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.,* [93, 29, 95, 61, 62, 72]). One of the most prominent lines of works (see [81] for a recent survey) considers anonymous attribute-based credentials (ABCs) [30], 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 [62]. Another related notion is that of delegatable anonymous credentials (DAC) [28, 16, 9, 90], which extend ACs to allow for a delegator to issuer 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) [17, 41, 40, 89] with [41] and [40] 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 [93, 63] for DAC not work as [93] has public keys in $(\mathbb{G}_2)^2$ with messages in $\mathbb{Z}_p$. Similarly, [63] 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 [42], 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 [93]. Derler and Slamanig present a variant of PS signatures with publicly randomizable keys [51]. 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 [51] 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: $\mathsf{pk}' = (\hat{X}', \hat{Y}') = \hat{X}\hat{Y}^{\rho_1}$ where the original key was $\mathsf{pk} = \hat{X}\hat{Y}$. 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

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

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

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

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

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

We also observe that this scheme can *only sign messages in* $\mathbb{Z}_p$ and so it cannot be used with GS proofs [68] (see Ghadafi signatures below).

Ghadafi signatures [63]. 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 [68]. 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 [51]). 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 [51]. 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 [3]. Similar to PS signatures and Ghadafi signatures, we can use the variant in [51] 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 [39, 46]. 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 [69]. 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 [51]. 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 [7, 42, 41] 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 [42], 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 $\boldsymbol{Adv}^{\alpha-\mathsf{UnfEquiv}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) := Pr\left[\boldsymbol{Exp}^{\alpha-\mathsf{UnfEquiv}}_{\mathsf{SWRK},\mathcal{A}}(\lambda) \Rightarrow \mathsf{true}\right] \leq \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\alpha-\mathsf{UnfEquiv}}_{\mathsf{SWRK},\mathcal{A}}(\lambda)$ is shown in Fig. 13.*

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

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

Oracle $\mathsf{Rand}()$
$\rho \leftarrow_\$ \mathcal{KR}$
$\Sigma_2 \leftarrow \Sigma_2 \cup \{\rho\}$
return $\rho$
```

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

C Updatable Signature Definitions

We present the unforgeability and unlinkability definitions from [39] 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 [39]) 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. 20) 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 [39]). *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 $\boldsymbol{Adv}^{\mathsf{us-euf-cma}}_{\Gamma,\mathcal{A}}(\lambda) := Pr\left[\boldsymbol{Exp}^{\mathsf{us-euf-cma}}_{\Gamma,\mathcal{A}}(\lambda,q)\right] = \epsilon(\lambda)$, where the experiment $\boldsymbol{Exp}^{\mathsf{us-euf-cma}}_{\Gamma,\mathcal{A}}(\lambda,q)$ is shown in Fig. 14*[7]

Definition 20 (Unlinkable updates under chosen-message attacks (US-UU-CMA) [39]). *A signature scheme, Γ, has unlinkable updates under chosen-message attacks, if for every* PPT *adversary $\mathcal{A}$, the advantage function defined by $\boldsymbol{Adv}^{\mathsf{us-uu-cma}}_{\Gamma,\mathcal{A}}(\lambda) := 2\cdot Pr\left[\boldsymbol{Exp}^{\mathsf{us-uu-cma}}_{\Gamma,\mathcal{A}}(\lambda,q) - 1\right] = \epsilon(\lambda)$, where $\boldsymbol{Exp}^{\mathsf{us-uu-cma}}_{\Gamma,\mathcal{A}}(\lambda,q)$ is shown in Fig. 15.*

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

[7] Since our focus here is on signatures, we do not consider the updatable MACs from [39], *i.e.,* we can remove the verification oracle (Ver').

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

Oracle $\mathsf{Sign}(e', m)$
    if $e' > e$ : return $\perp$
    $S = S \cup (e', m)$;
    return $\mathsf{Sign}(\mathsf{sk}_{e'}, m)$

Oracle $\mathsf{Next}(1^\lambda)$
    $(\mathsf{pk}_{e+1}, \mathsf{sk}_{e+1}, \rho_{e+1}) = \mathsf{Next}(\mathsf{pk}_e, \mathsf{sk}_e)$
    $I = I \cup (\mathsf{pk}_{e+1}, \mathsf{sk}_{e+1}, \rho_{e+1}); e := e + 1$
    return $\mathsf{pk}_e$

Oracle $\mathsf{Update}(e', m, \sigma)$
    if $e' > e$ : return $\perp$
    if $\mathsf{Verify}(\mathsf{pk}_{e'}, m, \sigma) \neq 1$ return $\perp$
    $S = S \cup (e'+1, m)$
    return $\mathsf{Update}(\rho_{e'+1}, m, \sigma)$

Oracle $\mathsf{CorruptKey}(e')$
    if $e' > e$ : return $\perp$
    $K = K \cup \{e'\}$
    return $\mathsf{sk}_{e'}$

Oracle $\mathsf{CorruptToken}(e')$
    if $e' > e$ : return $\perp$
    $T = T \cup \{e'\}$
    return $\rho_{e'}$
```

Fig. 14. Unforgeability experiment from [39].

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

Fig. 15. Unlinkable updates under chosen mesaage attack game from [39].