Ring Signatures with User-Controlled Linkability

Dario Fiore[1], Lydia Garms[1,2](✉), Dimitris Kolonelos[1,3], Claudio Soriente[4], and Ida Tucker[5]

[1] IMDEA Software Institute, Madrid, Spain
[2] Keyless Technologies Limited, London, UK
lydia.garms@keyless.io
[3] Universidad Politecnica de Madrid, Madrid, Spain
[4] NEC Laboratories Europe, Heidelberg, Germany
[5] Zondax AG, Zug, Switzerland

Abstract. Anonymous authentication primitives, e.g., group or ring signatures, allow one to realize privacy-preserving data collection applications, as they strike a balance between authenticity of data being collected and privacy of data providers. At PKC 2021, Diaz and Lehmann defined group signatures with User-Controlled Linkability (UCL) and provided an instantiation based on BBS+ signatures. In a nutshell, a signer of a UCL group signature scheme can link any of her signatures: linking evidence can be produced at signature time, or after signatures have been output, by providing an explicit linking proof.

In this paper, we introduce Ring Signatures with User-Controlled Linkability (RS-UCL). Compared to group signatures with user-controlled linkability, RS-UCL require no group manager and can be instantiated in a completely decentralized manner. We also introduce a variation, User Controlled and Autonomous Linkability (RS-UCAL), which gives the user full control of the linkability of their signatures.

We provide a formal model for both RS-UCL and RS-UCAL and introduce a compiler that can upgrade any ring signature scheme to RS-UCAL. The compiler leverages a new primitive we call Anonymous Key Randomizable Signatures (AKRS)—a signature scheme where the verification key can be randomized—that can be of independent interest. We also provide different instantiations of AKRS based on Schnorr signatures and on lattices. Finally, we show that an AKRS scheme can additionally be used to construct an RS-UCL scheme.

1 Introduction

Group signatures [CvH91, BMW03, BSZ05, BCC+16] and ring signatures [RST01] allow users to sign messages, while providing anonymity of signers and unlinkability of signatures. Group signatures require a central entity that manages group membership, whereas ring signatures allow a signer to choose an arbitrary ring of public keys, and sign on behalf of that ring.

V. Atluri et al. (Eds.): ESORICS 2022, LNCS 13555, pp. 405–426, 2022.
https://doi.org/10.1007/978-3-031-17146-8_20

Many application scenarios do not require full anonymity/unlinkability, and may actually ask for mechanisms to identify the signer or link signatures produced by the same party. Group signatures feature an opening authority that can de-anonymize signers and thereby test if two signatures have the same signer.

Recently, researchers have proposed more flexible linkability options for group signature schemes, where even less power is entrusted to the opener. In [HLC+11,HLC+13,SSU14,KTY04] group signatures with an authority who can test whether two signatures have the same signer but cannot open signatures were introduced. In [GL19,FGL21], group signatures are originally unlinkable, but later on can be converted to linkable signatures by an oblivious "converter". Group signatures with message-dependant opening [SEH+12,OSEH13, LJ14,LMN16] introduce an additional entity, the admitter, who can specify messages such that the corresponding signatures can be opened. Group signatures with certified limited opening [ZWC19] introduce a certifier, instead of an admitter, who can certify a particular opener to allow them to open signatures on messages within a particular context. Abe et al., [ACHO13] use "public-key anonymous tag system" to build a traceable signature scheme. In all these lines of work, linking is performed by a trusted party, and signers have no say in which of their signatures can be linked together. Another line of work [BCC04,BFG+13,CDL16b,CDL16a] considers scenarios where a so-powerful authority is undesirable, and achieves linkability by including a one-way function of the signing key and a "scope" chosen by the signer—also known as "pseudonym"—in each signature, so that two signatures with the same pseudonym can be trivially linked.

Pseudonym-based linkability, however, require signers to decide at signature time whether their signatures should be ever linked: two signatures using different scopes – hence, with different pseudonyms – would be unlinkable by definition. Recently, Diaz and Lehmann [DL21] introduced group signatures with User-Controlled Linkability (UCL) that provide pseudonym-based linkability (labelled "implicit" linkability), but also allow a signer to link any set of her signatures generated with the same linking secret, even if those had different pseudonyms (labelled "explicit" linkability). This linking model turns useful in applications where authenticated data is collected in anonymous fashion but, later on, one may be interested to link specific data items. For example, in smart-metering applications, energy consumptions may be collected in a fully anonymous way, while, at a later time, a user may want to link her measurements to, e.g., receive tailored offers from the energy providers. Similarly, connected vehicles can anonymously report their mobility traces but, at a later time, a driver may want to link her reports so to obtain discounts from insurance companies.

Our Contributions. In this paper, we continue the study of UCL but focus on ring signatures. Compared to group signatures, a ring signature scheme enables fully decentralized applications as no group manager is needed. Previous linkable ring signatures [LWW04,SALY17] solely allow anyone to link two signatures by the same signer on the same ring.

Our first contribution is the formalization of UCL in ring signatures. We introduce the first formal model for ring signatures with UCL allowing for both implicit and explicit linkability, as in [DL21]. Next, we introduce ring signatures with *User Controlled Autonomous Linkability* (UCAL) to give signers full control over the linkability of their signatures. Different from UCL, UCAL does not use the signing key to create pseudonyms, but instead uses a "linking secret" which can be re-used or chosen afresh. A fresh linking secret ensures that signatures cannot be linked (i.e., via implicit linkability) even if they have the same scope. Of course, a signer can use the same linking secret on multiple signatures with the same scope so to provide implicit linkability. Ultimately, a signer can prove linkability of any set of her signatures generated with the same linking secret, even if those had different pseudonyms. Further, using different linking secrets ensures that past signatures cannot be linked even if the signer is corrupted, providing a form of forward anonymity.

As a second contribution we propose constructions of UCL and UCAL ring signatures. To do so, we introduce a new cryptographic primitive that we label *Anonymous Key Randomizable Signatures* (AKRS) that may be of independent interest. AKRS is essentially a signature scheme where public keys can be re-randomized, while maintaining the correspondence to the same secret key, so that a randomized public key cannot be linked to the original one. However, by using the secret, the signer can prove that multiple public keys are all randomized versions of the original one. The primitive is in a similar spirit to Signatures with Flexible Public Key [BHKS18], which however is not fully suitable for our requirements. We elaborate more on the differences in Sect. 3.

We show that AKRS can be used to upgrade *any ring signature scheme* to UCAL. In particular, we use an AKRS public key that has been re-randomised with the scope as the pseudonym for the (ring) signature. By using a public key corresponding to the same AKRS secret key and scope, we provide implicit linkability. Otherwise, the signer may use a different AKRS secret key to make two signatures unlinkable even on the same scope. At a later time, the signer can use her AKRS secret key to link the pseudonyms of a set of ring signatures, thereby proving that all such signatures are linked. Notably, our construction does not modify the original ring signature public keys and thus can be used to upgrade to UCAL an already up-and-running system using ring signatures.

We also show how to use AKRS, along with a NIZK, to build a UCL ring signature. The linking mechanism is obtained using the AKRS in a similar way to the UCAL construction. The difference is that, for UCL the AKRS secret key is used as the ring signature secret. Therefore, to sign a message with some scope, in addition to using the AKRS secret key to compute the pseudonym, we also add a non-interactive signature of knowledge [CS97] that the secret used to derive the pseudonym corresponds to one of the public keys in the ring.

Finally, we propose two instantiations of AKRS: one based on Schnorr's signatures in prime order groups, and one based on Lyubashevsky's signatures [Lyu12] on lattices. Compiling our AKRS with a ring signature scheme we get UCAL ring signatures with minimal overhead: one additional Schnorr or Lyubashevsky signature, respectively. In contrast to previous works (on group signatures) [DL21], the constructions are generic and can bootstrap any arbitrary ring signature scheme to a UCAL one. For instance we can achieve UCAL without pairings.

Related Work. There is an extensive line of work studying and constructing efficient ring signatures from various settings and assumptions, with the today's state-of-the art achieving signatures of size logarithmic in the size of the ring [GK15,BCC+15,LPQ18,LRR+19,YSL+20,YEL+21].

Park and Sealfon [PS19] proposed the related notions of (un)Claimable and (un)Repudiable Ring Signatures. Claimability states that one can always claim a signature after she signed it, while Repudiability states the opposite, that one can always repudiate a signature she did not sign. Claimability is a notion relevant to user-controlled linkability, although it is weaker: a signer can claim a signature by linking it to a signature of a dummy message on-the-fly. The inverse, achieving user-controlled linkability from claimability, does not apply.

The idea of linking anonymous signatures of a user by using a Pseudorandom Function (PRF) has been used in the past. A user can additionally sign a random value together with its PRF output, where the seed is kept by the signer. Then, a NIZK proving knowledge of the (same) seed can be used to link signatures. However, this linking mechanism provides no succinctness: the linking proof typically grows with the number of linked signatures. AKRS generalizes this linking mechanism and guarantees succinctness; we note that an AKRS may also be instantiated with a PRF and a (succinct) NIZK.

2 Ring Signatures with User-Controlled Linkability

2.1 Standard Linkability

As in [DL21], we consider two types of linkability:

Implicit linkability: Signatures are accompanied by a pseudonym, generated by the user for a particular scope. Re-using the same scope leads to the same pseudonym, making all signatures with the same scope linkable. Signatures with different scopes cannot be linked, except via explicit link proofs.

Explicit linkability: A user can prove that she created a set of previously generated signatures, i.e. link the signatures in the set.

Definition 1 (RS-UCL). *A* ring signature scheme with user controlled linkability *(RS-UCL) is a tuple of PPT algorithms* $(\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Link}, \mathsf{VerifyLink})$*, satisfying* correctness*,* anonymity *(Definition 3),* unforgeability *(Definitions 4 and 6) and* non-frameability *(Definitions 7 and 8); and with the following syntax:*

$\mathsf{KGen}(1^\lambda)$ *on input a security parameter, outputs a signing key* sk *and a verification key* vk*.*

$\mathsf{Sig}(\mathsf{sk}, R, m, \mathsf{scp})$ *signs a message m w.r.t. scope* scp *via secret signing key* sk *for set of verification keys (called* the ring*)* $R = \{\mathsf{vk}_1, \ldots, \mathsf{vk}_n\}$*. The output is a pseudonym* nym *and a ring signature* σ*.*

$\mathsf{Vf}(\Sigma)$ *on input a signature tuple* $\Sigma = (m, \mathsf{scp}, R, \sigma, \mathsf{nym})$ *for ring* $R = \{\mathsf{vk}_1, \ldots, \mathsf{vk}_n\}$*, returns 1 if* σ *and* nym *are valid for message* m *and scope* scp *w.r.t.* R*, and 0 otherwise.*

$\mathsf{Link}(\mathsf{sk}, \mathsf{lm}, \boldsymbol{\Sigma})$ *on input a set of signature tuples* $\Sigma = \{\Sigma_i\}_{i\in[n]}$*, a user secret key* sk*, and a linking message* lm *(can be used to ensure freshness of the proof), outputs a proof* π_l *that these signatures are linked, or the error symbol* $\perp$*.*

$\mathsf{VerifyLink}(\mathsf{lm}, \boldsymbol{\Sigma}, \pi_l)$ *returns 1 if* π_l *is a valid proof that* $\boldsymbol{\Sigma} = \{\Sigma_i\}_{i\in[n]}$ *were produced by the same signer for link message* lm*, and 0 otherwise.*

2.2 Autonomous Linking

We also introduce the notion of ring signatures with user controlled, and *autonomous* linking (RS-UCAL). This variant allows users to chose the linking secret independently of the signing key. It hence gives users the liberty of choosing which of their signatures should be linkable in the future. We express this feature via an additional algorithm $\mathsf{GenLinkSec}$ which outputs a linking secret ls. Now both the signing algorithm and the linking algorithm should input both the signing secret sk and the linking secret ls.

Definition 2. *A* ring signature scheme with user controlled autonomous linking *(RS-UCAL) is a tuple of PPT algorithms* $(\mathsf{KGen}, \mathsf{GenLinkSec}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Link}, \mathsf{VerifyLink})$*, where* $\mathsf{KGen}, \mathsf{Vf}, \mathsf{VerifyLink}$ *have the same syntax as an RS-UCL, and:*

$\mathsf{GenLinkSec}(1^\lambda)$ *takes input a security parameter, and outputs a linking secret* ls*.*

$\mathsf{Sig}(\mathsf{sk}, \mathsf{ls}, R, m, \mathsf{scp})$ *as in a RS-UCL scheme, only with additional input a linking secret* ls*.*

$\mathsf{Link}(\mathsf{ls}, \mathsf{lm}, \boldsymbol{\Sigma})$ *as in an RS-UCL scheme, only with input a linking secret* ls *instead of the secret key.*

An RS-UCAL scheme satisfies correctness anonymity (Definition 3), unforgeability (Definitions 5 and 6) and non-frameability (Definitions 7 and 8).

Text which is highlighted in blue only occurs for a RS-UCAL scheme. Text which is highlighted in green only occurs for a RS-UCL scheme.

2.3 Correctness

An RS-UC(A)L scheme should satisfy both verification correctness, which is equivalent to correctness for standard ring signatures, and linking correctness which ensures the correctness of the explicit linking algorithm Link. We give the full definitions in the full version of this paper.

2.4 Security Model

For privacy, signatures must not reveal anything about the signer's identity beyond what was intended by her (**anonymity**). Security is expressed through both **unforgeability**, which ensures that an adversary cannot forge a signature on behalf of a ring they are not a member of or forge a link proof for signatures they did not generate, and **non-frameability**, which states that an honest user cannot be framed by the adversary, so that signatures of an honest user are (implicitly or explicitly) linkable to signatures that she has not generated.

Oracles and State. All oracles are parametrised by a list of keys pairs $\{(\mathsf{pk}_i, \mathsf{sk}_i)\}_{i\in[n]}$ and linking secrets $\{\mathsf{ls}_i\}_{i\in[n]}$, where $n = \mathsf{poly}(\lambda)$, $k = \mathsf{poly}(\lambda)$. In Fig. 1 we describe the global state variables that all oracles can access.

Corruption oracle Corr takes as input an index $i \in [n]$, adds i to the list of corrupted indices $I \leftarrow I \cup \{i\}$, and outputs the randomness w_i used to compute key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$.

Linking secret oracle OLS takes as input an index $i \in [k]$, adds i to the list of corrupted indices $J \leftarrow J \cup \{i\}$, and outputs the randomness w'_i used to compute linking secret ls_i.

Signing oracle $\mathsf{OSign}^{\mathsf{ucl\text{-}r}}$**:** takes as input a set R, an index $i \in [N]$, an index $j \in [k]$, a message m and a scope scp computes $(\mathsf{nym}, \sigma) \leftarrow \mathsf{Sig}(\mathsf{sk}_i, \mathsf{ls}_j, R \cup \mathsf{vk}_i, m, \mathsf{scp})$, adds $\Sigma := (m, \mathsf{scp}, R, \sigma, \mathsf{nym})$ to the list of signatures signed by user index i with linking secret index j: $\mathsf{SIG}[i, j] \leftarrow \mathsf{SIG}[i, j] \cup \{\Sigma\}$, and returns (nym, σ).

Linking oracle OLink**:** Allows the adversary to obtain link proofs for signatures of its choice. On input an index $i \in [n]$, an index $i \in [k]$, a linking message lm and a set of tuples $\boldsymbol{\Sigma} = \{\Sigma = (m, \mathsf{scp}, R, \sigma, \mathsf{nym})\}$, this oracle adds $(\mathsf{lm}, \boldsymbol{\Sigma})$ to the list of link proofs produced for index i: $\mathsf{LNK}[i] \leftarrow \mathsf{LNK}[i] \cup \{(\mathsf{lm}, \boldsymbol{\Sigma})\}$, and returns $\pi_l \leftarrow \mathsf{Link}(\mathsf{sk}_i, \mathsf{ls}_i, \mathsf{lm}, \boldsymbol{\Sigma})$.

Challenge signing oracle $\mathsf{Ch-Sign}_b$**:** Allows $\mathcal{A}$ to get signatures for challenge user index i_b with linking secret index j_b. On input a ring R, a message m, and a scope scp, $\mathsf{Ch-Sign}_b$ computes $(\mathsf{nym}, \sigma) \leftarrow \mathsf{Sig}(\mathsf{sk}_{i_b}, \mathsf{ls}_{j_b}, R \cup \{\mathsf{vk}_{i_0}, \mathsf{vk}_{i_1}\}, m, \mathsf{scp})$, sets $\Sigma := (m, \mathsf{scp}, R, \sigma, \mathsf{nym})$, adds Σ to the list of queried challenge signatures $\mathsf{CSIG} \leftarrow \mathsf{CSIG} \cup \{\Sigma\}$, and returns (nym, σ).

Challenge linking oracle $\mathsf{Ch-Link}_b$**:** Allows $\mathcal{A}$ to get link proofs for a challenge index i_b/j_b. Precisely, on input a linking message lm and a set of tuples $\boldsymbol{\Sigma} = \{\Sigma = (m, \mathsf{scp}, R, \sigma, \mathsf{nym})\}$, it adds $(\mathsf{lm}, \boldsymbol{\Sigma})$ to the list of challenge link proofs $\mathsf{CLNK} \leftarrow \mathsf{CLNK} \cup \{(\mathsf{lm}, \boldsymbol{\Sigma})\}$, and returns $\pi_l \leftarrow \mathsf{Link}(\mathsf{sk}_{i_b}, \mathsf{ls}_{j_b}, \mathsf{lm}, \boldsymbol{\Sigma})$.

Variable	Content
I	List of corrupted indices (queried to Corr)
J	List of corrupted indices (queried to OLS)
i_b	Challenge user index in $\mathsf{anon\text{-}b}$. Ignored in the other games
j_b	Challenge linking secret index in $\mathsf{anon\text{-}b}$. Ignored in the other games
$\mathsf{SIG}[i]$	Signature tuples produced by OSign for user index i
CSIG	Signature tuples produced by $\mathsf{Ch-Sign}_b$ for challenge user index i_b
$\mathsf{LNK}[i]$	Link queries sent to OLink for user index i
CLNK	Link queries made to $\mathsf{Ch-Link}_b$

Fig. 1. Global state variables and their contents.

Helper Algorithm. As in [DL21], we introduce the helper algorithm Identify. In this case we do not need to require the existence of Identify, because there is no trusted issuer in the ring signature setting. Therefore, user secret keys do not need to be extracted from join protocols as in the group signature setting. Identify here is just defined for notational simplicity, and can be achieved from the user controlled linkability functionality.

```
$\mathsf{Identify}(\mathsf{sk}, \mathsf{pk}, \mathsf{ls}, \Sigma = (m, \mathsf{scp}, R, \sigma, \mathsf{nym}))$
$(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KGen}(1^\lambda)$, $(\mathsf{nym}', \sigma') \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{ls}, \{vk\}, m, \mathsf{scp})$
$\textbf{if } \mathsf{nym}' = \mathsf{nym} \quad \textbf{return } 1 \quad \textbf{else return } 0$
```

Anonymity. Anonymity ensures that an adversary cannot figure out which of two (honest) challenge users generated a given signature. In formalizing this notion, one must be careful to exclude trivial wins leveraging user-controlled linkability (see the comments in the security experiment).

Definition 3 (Anonymity). *An RS-UCAL scheme satisfies adaptive anonymity against adversarially chosen keys if, for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, and any* $n = \mathsf{poly}(\lambda)$*,* $k = \mathsf{poly}(\lambda)$*, it holds that* $|\Pr[\mathsf{Exp}^{\textit{anon-1}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k) = 1] - \Pr[\mathsf{Exp}^{\textit{anon-0}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k) = 1]|$ *is negligible in* λ.

```
Experiment :$\mathsf{Exp}^{\textit{anon-b}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k)$
$\textbf{for } i = 1, \ldots, n \quad (\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
$\textbf{for } i = 1, \ldots, k \quad \mathsf{ls}_i \leftarrow \mathsf{GenLinkSec}(1^\lambda)$
$(i_0, i_1, j_0, j_1,, \mathsf{state}) \leftarrow \mathcal{A}_1^{\mathsf{OSign}^{\textit{ucl-r}}, \mathsf{OLink}, \mathsf{Corr}, \mathsf{OLS}}(\mathsf{choose}, \mathsf{vk}_1, \ldots, \mathsf{vk}_\mathsf{n})$
$d \leftarrow \mathcal{A}_2^{\mathsf{OSign}^{\textit{ucl-r}}, \mathsf{OLink}, \mathsf{Ch-Sign}_b, \mathsf{Ch-Link}_b, \mathsf{Corr}, \mathsf{OLS}}(\mathsf{guess}, \mathsf{state})$
// Exclude trivial wins via implicit linking: a signature for scope scp was queried to $\mathsf{Ch} - \mathsf{Sign}_b$ and to OSign for index $i_0$ or $i_1$
$\textbf{if } \exists \mathsf{scp} \ s.t.\ (*, \mathsf{scp}, *, *, *) \in \textit{CSIG} \wedge (*, \mathsf{scp}, *, *, *) \in \textit{SIG}[i_0, j_0] \cup \textit{SIG}[i_1, j_1]$
    $\textbf{thenreturn } \perp$
// Exclude trivial wins via explicit linking: both challenge and non challenge signatures were queried to $\mathsf{Ch} - \mathsf{Link}_b$ or to OLink
$\textbf{if } \exists \boldsymbol{\Sigma} \ s.t.\ (\boldsymbol{\Sigma} \cap \textit{CSIG} \neq \emptyset \wedge (*, \boldsymbol{\Sigma}) \in \textit{LNK}[*])$
 $\vee (\boldsymbol{\Sigma} \cap (\textit{SIG}[i_0, j_0] \cup \textit{SIG}[i_1, j_1]) \neq \emptyset \wedge (*, \boldsymbol{\Sigma}) \in \textit{CLNK}) \quad \textbf{thenreturn } \perp$
$\textbf{if } \{i_0, i_1\} \cap I \neq \emptyset \textbf{ thenreturn } \perp$
$\textbf{if } \{j_0, j_1\} \cap J \neq \emptyset \textbf{ thenreturn } \perp$
    $\textbf{else return } d$
```

Unforgeability. For both RS-UCL and RS-UCAL we must ensure that only ring members can sign. However, for RS-UCL, we must also prevent signers from outputting multiple unlinkable signatures on the same scope. For RS-UCL one captures both by defining an attack as the generation of more signatures with different pseudonyms than corrupted users in the ring, using the same scope. This is meaningless for RS-UCAL where a single corrupted user can generate many linking secrets.

Definition 4 (Signature unforgeability). *An RS-UCL scheme satisfies signature unforgeability if for any PPT adversary* $\mathcal{A}$*, and any* $n = \mathsf{poly}(\lambda)$*, it holds that* $\Pr[\mathsf{Exp}^{\textit{sig-uf}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n) = 1]$ *is negligible in* λ.

```
Experiment :$\mathsf{Exp}^{\textit{sig-uf}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n)$

for $i = 1, \dots, n$   $(\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\{\Sigma_1^*, \cdots, \Sigma_m^*\}, R^*) \leftarrow \mathcal{A}^{\mathsf{OSign}^{\textit{ucl-r}},\mathsf{OLink},\mathsf{Corr}}(\mathsf{vk}_1, \dots, \mathsf{vk}_n)$
$\textit{Parse}\ \Sigma_j^* = (m_j^*, \mathsf{scp}_j^*, R_j, \sigma_j^*, \mathsf{nym}_j^*)\ \textit{for}\ j \in [m]$
if $\exists j \in [m]\ \textit{s.t.}\ R_j \neq R^*$   return $0$
$m^* \leftarrow |R^* \backslash \{\mathsf{vk}_i\}_{i \in [n] \backslash I}|$
if $\textit{the following conditions all hold}$
  $1. m > m^*$ // more signatures are output than the corrupted users in the ring
  $2. \forall j_1, j_2 \in [m]$   $\mathsf{scp}_{j_1}^* = \mathsf{scp}_{j_2}^*$ and $\mathsf{nym}_{j_1}^* \neq \mathsf{nym}_{j_2}^*$ // all signatures are unlinked
  $3. \forall j \in [m]$   $\mathsf{Vf}(\Sigma_j^*) = 1$
  $4. \forall i \in [n], j \in [m]$   $(m_j^*, \mathsf{scp}_j^*, R^*, *, *) \notin \mathit{SIG}[i]$
    thenreturn $1$
else return $0$
```

As mentioned earlier, Definition 4 is too strong for RS-UCAL. Indeed, the autonomous linking property allows parties to change the linking secret as frequently as desired. Hence an adversary which wants to output many unlinkable secrets could simply keep changing the linking secret. For such schemes we adopt a similar notion of (signature) unforgeability as that of standard ring signatures, with the difference that the adversary also has access to a linking oracle. We give the full experiment in Appendix B.

Definition 5 (Signature unforgeability with autonomous linking). *An RS-UCAL scheme satisfies signature unforgeability if, for any PPT adversary $\mathcal{A}$, and any $n = \mathsf{poly}(\lambda)$, $k = \mathsf{poly}(\lambda)$, it holds that* $\Pr[\mathsf{Exp}^{\textit{sig-uf-al}}_{\mathsf{UCAL},\mathcal{A}}(1^\lambda, n, k) = 1]$ *is negligible in λ.*

We additionally define link unforgeability for both the RS-UCL and RS-UCAL schemes, which ensures that the linking proof is unforgeable.

Definition 6 (Link unforgeability). *An RS-UCAL scheme satisfies link unforgeability if, for any PPT adversary $\mathcal{A}$, and any $n = \mathsf{poly}(\lambda)$, $k = \mathsf{poly}(\lambda)$, the following is negligible in λ:* $\Pr[\mathsf{Exp}^{\textit{link-uf}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k) = 1]$.

```
Experiment :$\mathsf{Exp}^{\textit{link-uf}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k)$

for $i = 1, \dots, n$   $(\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
for $i = 1, \dots, k$   $\mathsf{ls}_i \leftarrow \mathsf{GenLinkSec}(1^\lambda)$
$(\mathsf{lm}, \mathbf{\Sigma}, \pi_l) \leftarrow \mathcal{A}^{\mathsf{OSign}^{\textit{ucl-r}},\mathsf{OLink},\mathsf{Corr},\mathsf{OLS}}(\mathsf{vk}_1, \dots, \mathsf{vk}_n)$
$\mathsf{VerifyLink}(\mathsf{lm}, \mathbf{\Sigma}, \pi_l) = 0\ \textit{or}\ \exists i \in [n][k] : (\mathsf{lm}, \mathbf{\Sigma}) \in \mathit{LNK}[i]$   return $0$
if $\exists i \in [n] : \forall \Sigma \in \mathbf{\Sigma} : \Sigma \in \mathit{SIG}[i]\ \textit{and}\ i \notin I$   return $1$
if $\exists i \in [k] : \forall \Sigma \in \mathbf{\Sigma} : \Sigma \in \mathit{SIG}[\cdot, i]\ \textit{and}\ i \notin J$   return $1$
else return $0$
```

Non-frameability. Non-frameability guarantees that an honest user cannot be framed by the adversary, such that signatures of an honest user are linkable to signatures that she has not generated. We capture this in the implicit/explicit setting with signature/link non-frameability respectively. We give the full games in Appendix B. Roughly, signature non-frameability ensures that an adversary cannot output a valid signature that links to another signature generated by an uncorrupted user via the signing oracle. Link non-frameability ensures that an adversary cannot explicitly link two signatures that were either from different users or such that one of the two signatures is not from a user in the experiment.

Definition 7 (Signature non-frameability). *An RS-UCAL scheme satisfies signature non-frameability if, for any PPT adversary $\mathcal{A}$, and any $n = \mathsf{poly}(\lambda)$, $k = \mathsf{poly}(\lambda)$, the following is negligible in λ:* $\Pr[\mathsf{Exp}^{\textit{sign-frame}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k) = 1]$.

Definition 8 (Link non-frameability). *An RS-UCAL scheme satisfies link non-frameability if, for any PPT adversary $\mathcal{A}$, and any $n = \mathsf{poly}(\lambda)$, $k = \mathsf{poly}(\lambda)$, the following is negligible in λ:* $\Pr[\mathsf{Exp}^{\textit{link-frame}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k) = 1]$.

3 Anonymous Key Randomisable Signatures

We will now give the syntax and security requirements for a new primitive we call Anonymous Key Randomisable Signatures (AKRS), which is closely related to Signatures with Flexible Public Key (SFPK) [BHKS18].

Intuition. An SFPK scheme is a standard signature scheme that allows for public and secret keys to be re-randomised. These re-randomised keys are considered to be in the same equivalence class as the original key pair. Such re-randomised public keys, and signatures which verify for them, should not be linkable to the original key pair of their equivalence class. This is formalised by a class hiding requirement. Furthermore, during key generation, a trapdoor can be generated allowing to efficiently decide if public keys are in the same equivalence class.

At first sight, an SFPK scheme seems to allow transforming ring signatures into ring signatures with user controlled and autonomous linking. A user's key pair would be that of a standard ring signature, while their linking secret key would be an SFPK key pair, with the corresponding trapdoor. During signing, the user's pseudonym could be a re-randomisation of the SFPK public key with the randomness set to be the scope, so that the resulting public key is within the same equivalence class as the original keypair in their linking key. The signature should include a standard ring signature to ensure that the signature was output by a ring member and a SFPK signature valid under the public key in the pseudonym to prevent framing attacks.

However, in the SFPK class hiding requirement, which is necessary to ensure anonymity of the ring signature, the adversary does not get to see the randomness used to re-randomise the public key. In their game, the knowledge of this randomness allows to trivially win, as an adversary can re-run the randomisation algorithm itself. For the application to ring signatures described in the previous paragraph, this will not do, since the randomness is a public value: the scope.

On the other hand, in this application, the SFPK's secret key remains hidden: it is part of the linking secret, which is unknown to the anonymity adversary. Therefore, we modify SFPK to ensure that the secret key is necessary to generate public keys in the same equivalence class, thereby avoiding the aforementioned trivial attack, while allowing for the adversary to know the randomness.

A second issue in the above construction, is that signatures can only be explicitly linked by revealing the trapdoor, which would allow *all signatures* under the same linking key to be linked. One way around this is to add an additional functionality, proving, in zero-knowledge, that the pseudonyms in signatures are in the same equivalence class, using the trapdoor in the linking secret key. For efficiency, however, we chose a different approach: we allow for several key pairs in the same equivalence class to be accumulated into one key pair. Then the accumulated secret key could be used to sign a signature valid under the accumulated public key. A verifier can check that this signature is valid under an accumulated public key, resulting from the pseudonyms of all signatures being linked. This means the linking proof is only the size of an SFPK signature. However, explicitly linking signatures requires keeping track of each re-randomised secret key used in the linking key. We overcome this by re-randomising only public keys, but not secret keys, i.e., the secret key remains the same for all public keys in a given equivalence class. An accumulated public key will then correspond to the same secret key as the public keys it was built from, as long as the latter lie in the same equivalence class. Finally, observe that the secret key essentially fulfils the role of the trapdoor, as it allows to link public keys in the same equivalence class. We thus remove the trapdoor from our primitive's syntax.

We now formally define Anonymous Key Randomisable Signatures. The security properties required are given in Sect. 3.1 to 3.3.

Definition 9 (Anonymous Key Randomisable Signature (AKRS)). *An* anonymous key randomisable signature scheme *is a tuple of PPT algorithms* $(\mathsf{KGen}, \mathsf{ChgPK}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Accum})$*, satisfying* correctness *(Definition 10),* class hiding *(Definition 11),* existential unforgeability under chosen message attacks *(Definition 12) and* accumulation soundness *(Definition 13); and with the following syntax:*

$\mathsf{KGen}(1^\lambda)$ *on input a security parameter* 1^λ*, outputs a signing key* sk *and a public key* pk*.*

$\mathsf{ChgPK}(\mathsf{sk}, t)$ *on input a secret key* sk *and a tag* t*, outputs a new public key* pk' *in the same equivalence class.*

$\mathsf{Sig}(\mathsf{sk}, \mathsf{pk}, m)$ *on input a signing key* sk*, a public key* pk *and a message* m*, outputs a signature* σ*.*

$\mathsf{Vf}(\mathsf{pk}, m, \sigma)$ *on input a public key* pk*, a message* m *and a signature* σ*, the verification algorithm returns 1 if* σ *is a valid signature on* m *w.r.t.* pk*, and 0 otherwise.*

$\mathsf{Accum}((t_1, \mathsf{pk}_1), \cdots, (t_k, \mathsf{pk}_k))$ *on input* k *public keys* $\mathsf{pk}_1, \cdots, \mathsf{pk}_k$ *with respect to tags* $t_1, \cdots, t_k$*, outputs an accumulated public key* $\widetilde{\mathsf{pk}}$*.*

Definition 10 (Correctness). *Consider any positive integer k; any tags [BCC+15]$t_1, \cdots, t_k \in \{0,1\}^*$; let* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda)$*, for all* $i \in [k]$ $\mathsf{pk}'_i \leftarrow \mathsf{ChgPK}(\mathsf{sk}, t_i)$*, and* $\tilde{\mathsf{pk}} \leftarrow \mathsf{Accum}((t_1, \mathsf{pk}'_1), \cdots (t_k, \mathsf{pk}'_k))$*. An AKRS scheme is* correct *if for any message m, there exists a negligible function ϵ such that:*

$$\Pr\left[\begin{array}{c|c} \mathsf{Vf}(\mathsf{pk}, m, \sigma) = 1 & \sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{pk}, m), \\ \wedge \forall i \in [k],\ \mathsf{Vf}(\mathsf{pk}'_i, m, \sigma_i) = 1 & \forall i \in [k],\ \sigma_i \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{pk}'_i, m), \\ \wedge \mathsf{Vf}(\tilde{\mathsf{pk}}, m, \tilde{\sigma}) = 1 & \tilde{\sigma} \leftarrow \mathsf{Sig}(\mathsf{sk}, \tilde{\mathsf{pk}}, m) \end{array}\right] = 1 - \epsilon(\lambda).$$

If $\epsilon = 0$ then perfect correctness *is satisfied.*

3.1 Class Hiding

We ensure that an adversary cannot guess which of two public keys are re-randomised with a tag *chosen by the adversary* (through oracle OChgPK), even while they can obtain signatures on these public keys and the re-randomised keys (via the OSign oracle).

Definition 11 (Class Hiding). *An* anonymous key randomisable signature scheme *satisfies class hiding if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, [BCC+15]*$\Pr[\mathsf{Exp}^{\textit{class-hiding}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda) = 1]$ *is negligible in λ:*

```
Experiment :$\mathsf{Exp}^{\mathsf{class\text{-}hiding}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda)$
$b^* \leftarrow_\$ \{0,1\}$
for $b \in \{0,1\}$    $(\mathsf{sk}_b, \mathsf{pk}_b) \leftarrow \mathsf{KGen}(1^\lambda)$
$\mathsf{sk}_2 \leftarrow \mathsf{sk}_{b^*}, L_1 := \{\}, L_2 := \{\}$
$d \leftarrow \mathcal{A}^{\mathsf{OChgPK}_{L_1,L_2}, \mathsf{OSign}_{L_1,L_2}}(\mathsf{choose}, \mathsf{pk}_0, \mathsf{pk}_1)$
return $((d = b^*) \wedge (L_1 \cap L_2 = \emptyset))$
```

```
$\mathsf{OChgPK}_{L_1,L_2}(a, t)$
if $\mathsf{sk}_a = \perp$ thenreturn $\perp$
if $a \in \{0,1\}$ then$L_1 \leftarrow L_1 \cup \{t\}$
if $a = 2$ then$L_2 \leftarrow L_2 \cup \{t\}$
$\mathsf{pk}' \leftarrow \mathsf{ChgPK}(\mathsf{sk}_a, t)$
return $\mathsf{pk}'$
```

```
$\mathsf{OSign}_{L_1,L_2}(a, m, \mathsf{pk}', \{t_1, \cdots, t_m\})$
if $\mathsf{sk}_a = \perp$ thenreturn $\perp$
if $\{t_1, \cdots, t_m\} \neq *$ then
    if $a \in \{0,1\}$ then$L_1 \leftarrow L_1 \cup \{t_1, \cdots, t_m\}$
    if $a = 2$ then$L_2 \leftarrow L_2 \cup \{t_1, \cdots, t_m\}$
    if $\mathsf{pk}' \neq \mathsf{Accum}((t_1, \mathsf{ChgPK}(\mathsf{sk}_a, t_1)), \cdots, (t_m, \mathsf{ChgPK}(\mathsf{sk}_a, t_m)))$ thenreturn $\perp$
    $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}_a, \mathsf{pk}', m)$
if $t = *$ thenif $a = 2$ thenreturn $\perp$
    else $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}_a, \mathsf{pk}_a, m)$
return $\sigma$
```

3.2 Existential Unforgeability Under Chosen Message Attacks

We ensure that signatures cannot be forged for a public key in the equivalence class of an honest user, even when they can obtain multiple public keys in that equivalence class on tags of their choice. We give the full game in Appendix A.

Definition 12 (Existential Unforgeability under Chosen Message Attacks). *An* anonymous key randomisable signature scheme *satisfies existential unforgeability under chosen message attacks if for any PPT adversary $\mathcal{A}$,* $\Pr[\mathsf{Exp}^{\mathsf{euf-cma}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda) = 1]$ *is negligible in λ.*

3.3 Accumulation Soundness

This is a new requirement necessary due to the accumulation functionality. It must not be possible to produce a signature which verifies for an accumulated public key, if the public keys input to the accumulation algorithm do not belong to the same equivalence class.

Definition 13 (Accumulation Soundness). *An* anonymous key randomisable signature scheme *satisfies accumulation soundness if for any PPT adversary $\mathcal{A}$,* $\Pr[\mathsf{Exp}^{\mathsf{acc-sound}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda) = 1]$ *is negligible in λ.*

```
Experiment :$\mathsf{Exp}^{\mathsf{acc-sound}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda)$
for $i \in [k]$   $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\{(\hat{t}_i, \hat{pk}_i, \hat{m}_i, \hat{\sigma}_i)\}_{i\in[k+1,k+l]}, \mathcal{I}, \{t_i\}_{i\in\mathcal{I}}, m, \sigma) \leftarrow \mathcal{A}((\mathsf{sk}_1, \mathsf{pk}_1), \cdots, (\mathsf{sk}_k, \mathsf{pk}_k))$
if $\exists (i,j) \in \mathcal{I} \cup [k+1, k+l]$ such that $i \neq j$ and $t_i = t_j$   return 0
if $\mathcal{I} = \emptyset$   return 0
if $\exists i \in [k+1, k+l]$ s.t $\mathsf{Vf}(\hat{pk}_i, \hat{m}_i, \hat{\sigma}_i) = 0$   return 0
for $i \in \mathcal{I}$   $\mathsf{pk}'_i \leftarrow \mathsf{ChgPK}(\mathsf{sk}_i, t_i)$
$\tilde{\mathsf{pk}} \leftarrow \mathsf{Accum}(\{(t_i, \mathsf{pk}'_i)\}_{i\in\mathcal{I}}, \{(\hat{t}_i, \hat{\mathsf{pk}}_i)\}_{i\in[k+1,k+l]})$
if $\mathsf{Vf}(\tilde{\mathsf{pk}}, m, \sigma) = 0$   return 0
if $\exists (i,j) \in \mathcal{I}$ s.t. $\mathsf{sk}_i \neq \mathsf{sk}_j \vee \exists i \in [k+1, k+l]$ s.t. $\forall j \in \mathcal{I}, \mathsf{ChgPK}(sk_j, \hat{t}_i) \neq \hat{pk}_i$
    thenreturn 1
else return 0
```

4 Constructions for RS-UCL and RS-UCAL

4.1 A RS-UCAL Construction

We provide a generic construction, upgrading a standard ring signature scheme to a scheme with user controlled autonomous linking. We build upon a standard ring signature scheme $\mathsf{RS} := (\mathsf{KGen}, \mathsf{Sig}, \mathsf{Vf})$ as described formally in the full version of the paper and an AKRS $\mathsf{AKRS} = (\mathsf{KGen}, \mathsf{ChgPK}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Accum})$, as defined in Sect. 3.

Our RS-UCAL scheme UCAL, given in Fig. 2, works as follows. Key generation is simply the key generation for a standard ring signature, whereas the linking secret is the secret key for an AKRS. When signing, the pseudonym is the public key corresponding to the linking secret, randomised with respect to the scope, a one way function of the linking secret and scope as desired. We then included a standard ring signature to ensure that a non-member cannot sign on behalf of the ring, i.e. signature unforgeability. In order to prevent a signature non-frameability attack, where an adversary uses the pseudonym of an honest user's signature in their own signature, we also include an AKRS signature with respect to the pseudonym as the public key. In order to explicitly link signatures we make use of the accumulation functionality from AKRS. A set of signatures that were all generated with the same linking secret contain pseudonyms that can be accumulated. This accumulated public key can be used to sign an AKRS signature, which will be the link proof. Due to the accumulation soundness property and unforgeability of AKRS, link non-frameability and link unforgeability are ensured, respectively. Anonymity follows from the anonymity of standard ring signatures, and the class hiding property of the AKRS which ensures AKRS public keys and signatures cannot be linked based on equivalence class, and so UCAL signatures cannot be linked based on the linking secret.

We prove the following theorem that UCAL is a secure UCAL ring signature in the full version of the paper.

Theorem 1. *If* AKRS *is an anonymous key re-randomisable signature scheme and* RS *is a (standard) ring signature scheme, then* UCAL *is a ring signature scheme with user controlled autonomous linking.*

```
$\mathsf{KGen}(1^\lambda)$
    $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{RS.KGen}(1^\lambda)$
    return $(\mathsf{sk}, \mathsf{vk})$
$\mathsf{GenLinkSec}(1^\lambda)$
    $(\mathsf{ls}, \cdot) \leftarrow \mathsf{AKRS.KGen}(1^\lambda)$   return $\mathsf{ls}$
$\mathsf{Sig}(\mathsf{sk}, \mathsf{ls}, R, m, \mathsf{scp})$
    Parse $(\mathsf{vk}_1, \ldots, \mathsf{vk}_n) \leftarrow R$
    $\mathsf{nym} \leftarrow \mathsf{AKRS.ChgPK}(\mathsf{ls}, \mathsf{scp})$
    $\Omega \leftarrow \mathsf{RS.Sig}(\mathsf{sk}, R, m||\mathsf{scp}||\mathsf{nym})$
    $\Psi \leftarrow \mathsf{AKRS.Sig}(\mathsf{ls}, \mathsf{nym}, m||\mathsf{scp}||R||\Omega)$
    $\sigma \leftarrow (\Omega, \Psi)$
    return $(\mathsf{nym}, \sigma)$
$\mathsf{Vf}(m, \mathsf{scp}, R, (\Omega, \Psi), \mathsf{nym})$
    Parse $(\mathsf{vk}_1, \ldots, \mathsf{vk}_n) \leftarrow R$
    if $\mathsf{RS.Vf}(\Omega, R, m||\mathsf{scp}||\mathsf{nym}) = 0$
        thenreturn $0$
    if $\mathsf{AKRS.Vf}(\mathsf{nym}, m||\mathsf{scp}||R||\Omega, \Psi) = 0$
        thenreturn $0$
    return $1$
```

```
$\mathsf{Link}(\mathsf{ls}, \mathsf{lm}, \mathbf{\Sigma})$
    if Parse $\{(m_i, \mathsf{scp}_i, R_i, \sigma_i, \mathsf{nym}_i)\}_{i \in [k]} \leftarrow \mathbf{\Sigma}$
    if $\exists i, j \in [k], (i \neq j) \wedge (\mathsf{scp}_i = \mathsf{scp}_j)$
        thenreturn $\perp$
    if $\exists i \in [k], \mathsf{nym}_i \neq \mathsf{AKRS.ChgPK}(\mathsf{ls}, \mathsf{scp}_i)$
        thenreturn $\perp$
    if $\exists i \in [k], \mathsf{Vf}(m_i, \mathsf{scp}_i, R_i, \sigma_i, \mathsf{nym}_i) = 0$
        thenreturn $\perp$
    $\tilde{\mathsf{nym}} \leftarrow \mathsf{Accum}(\mathsf{nym}_1, \ldots, \mathsf{nym}_k)$
    $\pi_l \leftarrow \mathsf{AKRS.Sig}(\mathsf{ls}, \tilde{\mathsf{nym}}, \mathsf{lm}||\mathbf{\Sigma})$
    return $\pi_l$
$\mathsf{VerifyLink}(\mathsf{lm}, \mathbf{\Sigma}, \pi_l)$
    Parse $\{(m_i, \mathsf{scp}_i, R_i, \sigma_i, \mathsf{nym}_i)\}_{i \in [k]} \leftarrow \mathbf{\Sigma}$
    if $\exists i, j \in [k], (i \neq j) \wedge (\mathsf{scp}_i = \mathsf{scp}_j)$
        thenreturn $0$
    if $\exists i \in [k], \mathsf{Vf}(m_i, \mathsf{scp}_i, R_i, \sigma_i, \mathsf{nym}_i) = 0$
        thenreturn $0$
    $\tilde{\mathsf{nym}} \leftarrow \mathsf{Accum}(\mathsf{nym}_1, \ldots, \mathsf{nym}_k)$
    return $\mathsf{AKRS.Vf}(\tilde{\mathsf{nym}}, \mathsf{lm}||\mathbf{\Sigma}, \pi_l)$
```

Fig. 2. RS-UCAL from standard ring signatures and AKRS.

4.2 Construction for Ring Signatures with User Controlled Linkability

We provide a generic construction for a ring signature scheme with user controlled linking. We build upon a NIZK for the language $\mathcal{L} = \{(\mathsf{nym}, \mathsf{scp},$

$(\mathsf{vk}_i)_{i\in[n]};\mathsf{sk}) : \mathsf{nym} = \mathsf{AKRS.ChgPK}(\mathsf{sk},\mathsf{scp}) \wedge \bigvee_{i\in[n]}(\mathsf{vk}_i,\mathsf{sk}) = \mathsf{AKRS.KGen}(1^\lambda)\}$ and an AKRS $\mathsf{AKRS} = (\mathsf{KGen},\mathsf{ChgPK},\mathsf{Sig},\mathsf{Vf},\mathsf{Accum})$, as defined in Sect. 3.

Our RS-UCL scheme UCL, given in Fig. 3, works as follows. Key generation (instead of the linking secret generation) is now identical to key generation for an AKRS. When signing, the pseudonym, similarly to the UCAL construction, is the AKRS public key randomised with the scope and now with the ring signature secret key corresponding to the AKRS secret key. For the RS-UCL model, we additionally need to prove that the pseudonym was generated with a secret key corresponding to one of the public keys in the ring to ensure signature unforgeability. This ensures that if an adversary holds n keys in the ring, they can only generate n different pseudonyms and so output n unlinked signatures. We therefore attach a non-interactive zero-knowledge proof of knowledge that attests to this, which also fulfills the role of the ring signature in the UCAL construction. The AKRS signature from the UCAL construction is now also not needed to ensure signature non-frameability, because it is necessary to know the secret key corresponding to a pseudonym in order to generate the NIZK. Explicit linking can be done in exactly the same way as the UCAL construction with link non-frameability and link unforgeability satisfied in the same way. Anonymity similarly follows from the class hiding property of the AKRS, and now also from the zero knowledge property of the NIZK.

```
$\mathsf{KGen}(1^\lambda)$
  $(\mathsf{sk},\mathsf{vk}) \leftarrow \mathsf{AKRS.KGen}(1^\lambda)$
  return $(\mathsf{sk},\mathsf{vk})$
```

```
$\mathsf{Sig}(\mathsf{sk}, R, m, \mathsf{scp})$
  Parse $(\mathsf{vk}_1,\ldots,\mathsf{vk}_n) \leftarrow R$
  $\mathsf{nym} \leftarrow \mathsf{AKRS.ChgPK}(\mathsf{sk},\mathsf{scp})$
  $\sigma \leftarrow \mathsf{NIZK}\{\mathsf{sk} : \mathsf{nym} = \mathsf{AKRS.ChgPK}(\mathsf{sk},\mathsf{scp})$
   $\wedge \bigvee_{i\in[n]} (\mathsf{vk}_i,\mathsf{sk}) \in \mathsf{AKRS.KGen}(1^\lambda)\}(m)$
  return $(\mathsf{nym},\sigma)$
```

```
$\mathsf{Vf}(m,\mathsf{scp},R,\sigma,\mathsf{nym})$
  Parse $(\mathsf{vk}_1,\ldots,\mathsf{vk}_n) \leftarrow R$
  Verify NIZK $\sigma$ wrt statement $(\mathsf{nym},\mathsf{scp},m,$
  $(\mathsf{vk}_1,\ldots,\mathsf{vk}_n))$
  // Note $m$ was auxilary input to the NIZK $\sigma$
```

```
$\mathsf{Link}(\mathsf{sk},\mathsf{lm},\mathbf{\Sigma})$
  Parse $\{(m_i,\mathsf{scp}_i,R_i,\sigma_i,\mathsf{nym}_i)\}_{i\in[k]} \leftarrow \mathbf{\Sigma}$
  if $\exists i,j \in [k], (i \neq j) \wedge (\mathsf{scp}_i = \mathsf{scp}_j)$
    thenreturn $\perp$
  if $\exists i \in [k], \mathsf{nym}_i \neq \mathsf{AKRS.ChgPK}(\mathsf{sk},\mathsf{scp}_i)$
    thenreturn $\perp$
  if $\exists i \in [k], \mathsf{Vf}(m_i,\mathsf{scp}_i,R_i,\sigma_i,\mathsf{nym}_i) = 0$
    thenreturn $\perp$
  $\tilde{\mathsf{nym}} \leftarrow \mathsf{Accum}(\mathsf{nym}_1,\ldots,\mathsf{nym}_k)$
  $\pi_l \leftarrow \mathsf{AKRS.Sig}(\mathsf{sk},\tilde{\mathsf{nym}},\mathsf{lm}||\mathbf{\Sigma})$
  return $\pi_l$
```

```
$\mathsf{VerifyLink}(\mathsf{lm},\mathbf{\Sigma},\pi_l)$
  Parse $\{(m_i,\mathsf{scp}_i,R_i,\sigma_i,\mathsf{nym}_i)\}_{i\in[k]} \leftarrow \mathbf{\Sigma}$
  if $\exists i,j \in [k], (i \neq j) \wedge (\mathsf{scp}_i = \mathsf{scp}_j)$
    thenreturn $0$
  if $\exists i \in [k], \mathsf{Vf}(m_i,\mathsf{scp}_i,R_i,\sigma_i,\mathsf{nym}_i) = 0$
    thenreturn $0$
  $\tilde{\mathsf{nym}} \leftarrow \mathsf{Accum}(\mathsf{nym}_1,\ldots,\mathsf{nym}_k)$
  return $\mathsf{AKRS.Vf}(\tilde{\mathsf{nym}},\mathsf{lm}||\mathbf{\Sigma},\pi_l)$
```

Fig. 3. RS-UCL from AKRS and non-interactive zero knowledge proofs of knowledge.

We prove the following theorem that UCL is a secure UCL ring signature in the full version of the paper.

Theorem 2. *If* AKRS *is an anonymous key re-randomisable signature scheme and* NIZK *is a non-interactive zero knowledge proof of knowledge satisfying simulation sound extractability, then* UCL *is a UCL ring signature.*

5 AKRS Instantiations

5.1 Construction from Schnorr Signatures

Consider a group $\mathbb{G}$, of prime order q, generated by g; and hash functions $\mathsf{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}$, and $\mathsf{H}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q$. In Fig. 4, we recall the standard algorithms of the standard Schnorr signature scheme. We also introduce new algorithms ChgPK and Accum which augment the scheme to an AKRS.

Security Argument. We now provide some intuition as to why the augmented Schnorr signature given in Fig. 4 satisfies our requirements for an AKRS. Unforgeability is, modulo minor technical details, inherent from the unforgeability property of Schnorr Signatures. Class hiding follows from the fact that two public keys in the same equivalence class are of the form $(\mathsf{H}_1(t), \mathsf{H}_1(t)^x)$ and $(\mathsf{H}_1(t'), \mathsf{H}_1(t')^x)$, which is a DDH tuple. Therefore, linking public keys by equivalence class can be reduced to distinguishing DDH tuples. In the proof, signatures can be simulated without the secret key, assuming that H_2 is a random oracle. Accumulation soundness is the property that involves more novel techniques, in the design of our construction and its security analysis. The accumulation algorithm can be seen as a way to batch ℓ Schnorr statements with the same witness into a *succinct proof* that, to the best of our knowledge, is new. Roughly, accumulation soundness follows from the fact that signing with respect to an accumulated public key $(\prod \tilde{g}_i, \prod \tilde{h}_i)$ requires knowledge of the discrete logarithm of $\prod \tilde{h}_i$ base $\prod \tilde{g}_i$. Letting $\prod \tilde{g}_i = \prod \mathsf{H}_1(t_i) = g^{\sum \tilde{t}_i}$, where $\mathsf{H}_1(t_i) = g^{\tilde{t}_i}$ the adversary must know $\frac{\sum \mathsf{sk}_i \tilde{t}_i}{\sum \tilde{t}_i}$, which entails knowledge of the $\tilde{t}_i$, i.e. breaking the discrete logarithm. For lack of space we leave the formal proofs for the full version of the paper.

```
KGen($1^\lambda$)
  • $\mathsf{sk} := x \leftarrow_\$ \mathbb{Z}_q$
  • $\mathsf{vk} \leftarrow (g, g^x)$
  • Output
    $(\mathsf{sk}, \mathsf{vk})$
ChgPK($\mathsf{sk}, t$)
  • $\tilde{g} \leftarrow \mathsf{H}_1(t)$;
  • $\tilde{h} \leftarrow \tilde{g}^x$
  • Output $(\tilde{g}, \tilde{h})$
```

```
Sig($\mathsf{sk}, (\tilde{g}, \tilde{h}), m$)
  • $k \leftarrow_\$ \mathbb{Z}_q$, $r \leftarrow \tilde{g}^k$
  • $e \leftarrow \mathsf{H}_2(\tilde{g}||\tilde{h}||r||m)$
  • $s \leftarrow k - xe \bmod q$
  • Output $(s, e)$
Vf($(\tilde{g}, \tilde{h}), m, (s, e)$)
  • if $\tilde{g} = 1$ reject
  • $r_v \leftarrow \tilde{g}^s \tilde{h}^e$
  • $e_v \leftarrow \mathsf{H}_2(\tilde{g}||\tilde{h}||r_v||m)$
  • If $e = e_v$ then accept,
    else reject
```

```
Accum($(t_1, (\tilde{g}_1, \tilde{h}_1)), \ldots, (t_n, (\tilde{g}_n, \tilde{h}_n))$)
  • if $\exists i \in [n]$ s.t. $\tilde{g}_i \neq \mathsf{H}_1(t_i)$
    thenreturn $\perp$
  • $\tilde{g} \leftarrow \prod \tilde{g}_i$
  • $\tilde{h} \leftarrow \prod \tilde{h}_i$
  • Output $\mathsf{pk} := (\tilde{g}, \tilde{h})$
```

Fig. 4. Schnorr AKRS

Compiling to UCAL and UCL. Combined with any Ring Signature scheme the above AKRS gives a UCAL-Ring Signature with a very small overhead: for the signature size it is 1 additional Schnorr Signature, while all the extra computational costs are insignificant. Then, linking ℓ signatures requires a group multiplication of ℓ elements and a Schnorr Signature. Verifying the linking of ℓ signatures requires ℓ group multiplications and a Schnorr-Signature verification. For UCL the main efficiency overhead comes from the NIZK. Our Schnorr-based AKRS allows for the k-out-of-n NIZK by Attema et al. [ACF21] to be used (setting $k = 1$), which gives similar asymptotic performance to the state-of-the-art on Ring Signatures [GK15, BCC+15, LPQ18, LRR+19, YSL+20, YEL+21].

5.2 Lattice Construction

Our Lattice-based AKRS is based on the Fiat-Shamir signature scheme by Lyubashevsky [Lyu12], which can be seen as the Lattice analogue of Schnorr signatures. We show how to bootstrap this signature scheme to an AKRS. For the sake of simplicity we describe the scheme w.r.t. integer lattices (based on SIS). However, it extends normally to ideal lattices (based on ring-SIS). The construction is in Fig. 5. where in the above $D^{\mu}_{\boldsymbol{v},\sigma}(\cdot)$ is the discrete normal distribution over $\mathbb{Z}^{\mu}$ centered around $\boldsymbol{v} \in \mathbb{Z}^{\mu}$ ($D^{\mu}_{\sigma}(\cdot)$ centered around $\boldsymbol{v} = 0$ resp.) with standard deviation σ and $n, \mu, k, \sigma, M, \eta, d$ are parameters. We refer to [Lyu12] for details.

```
$\mathsf{KGen}(1^{\lambda})$
  • $\boldsymbol{S} \leftarrow_{\$} [-d, d]^{\mu \times k}$
    $\mathsf{sk} := \boldsymbol{S}$
  • $\boldsymbol{A} \leftarrow_{\$} \mathbb{Z}_q^{n \times \mu}$, $T = \boldsymbol{AS}$
    $\mathsf{vk} := (\boldsymbol{A}, \boldsymbol{T})$
  • Output $(\mathsf{sk}, \mathsf{vk})$
$\mathsf{ChgPK}(\mathsf{sk}, t)$
  • $\tilde{\boldsymbol{A}} \leftarrow \mathsf{H}_1(t)$;
  • $\tilde{\boldsymbol{T}} \leftarrow \tilde{\boldsymbol{A}}\boldsymbol{S}$
  • Output $(\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{T}})$

$\mathsf{Sig}(\mathsf{sk}, (\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{T}}), m)$
  • $\boldsymbol{k} \leftarrow_{\$} D^{\mu}_{\sigma}$, $\boldsymbol{r} \leftarrow \tilde{\boldsymbol{A}}\boldsymbol{k}$
  • $\boldsymbol{e} \leftarrow \mathsf{H}_2(\tilde{\boldsymbol{A}}||\tilde{\boldsymbol{T}}||\boldsymbol{r}||m)$
  • $\boldsymbol{s} \leftarrow \boldsymbol{Se} + \boldsymbol{k}$
  • Output $(\boldsymbol{s}, \boldsymbol{e})$ with probability
    $\Pr = \min\left\{\frac{D^{\mu}_{\sigma}(\boldsymbol{z})}{M D^{\mu}_{\boldsymbol{Se},\sigma}(\boldsymbol{s})}, 1\right\}$
  • Otherwise repeat.
$\mathsf{Vf}((\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{T}}), m, (\boldsymbol{s}, \boldsymbol{e}))$
  • Accept if:
    1. $\boldsymbol{e} = \mathsf{H}_2(\tilde{\boldsymbol{A}}||\tilde{\boldsymbol{T}}||\tilde{\boldsymbol{A}}\boldsymbol{s} - \tilde{\boldsymbol{T}}\boldsymbol{e}||m)$
    2. $\|\boldsymbol{s}\| \leq \eta\sigma\sqrt{\mu}$
  • else reject

$\mathsf{Accum}\left(\left(t_i, (\tilde{\boldsymbol{A}}_i, \tilde{\boldsymbol{T}}_i)\right)_{i=1}^{\ell}\right)$
  • **if** $\tilde{\boldsymbol{A}}_i \neq \mathsf{H}_1(t_i)$
    **thenreturn** $\perp$
  • $\tilde{\boldsymbol{A}} \leftarrow \sum_i \tilde{\boldsymbol{A}}_i$
  • $\tilde{\boldsymbol{T}} \leftarrow \prod_i \tilde{\boldsymbol{T}}_i$
  • Output $\mathsf{pk} := (\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{T}})$
```

Fig. 5. Lattice-based AKRS

Security and Parameters. For Class hiding to hold it is sufficient to show that $(\tilde{\boldsymbol{A}}_1, \tilde{\boldsymbol{A}}_1\boldsymbol{S}) \approx (\tilde{\boldsymbol{A}}_2, \tilde{\boldsymbol{A}}_2\boldsymbol{S}) \approx \ldots \approx (\tilde{\boldsymbol{A}}_{\ell}, \tilde{\boldsymbol{A}}_{\ell}\boldsymbol{S})$, where $\tilde{\boldsymbol{A}}_i \leftarrow_{\$} \mathbb{Z}_q^{n \times \mu}, \boldsymbol{S} \leftarrow_{\$} [-d, d]^{\mu \times k}$. If we apply the Leftover hash lemma [HILL99] to the hash function:

$$f(\boldsymbol{S}) = \begin{pmatrix} \tilde{\boldsymbol{A}}_1 & 0 & \ldots & 0 \\ 0 & \tilde{\boldsymbol{A}}_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \tilde{\boldsymbol{A}}_{\ell} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{S} \\ \boldsymbol{S} \\ \vdots \\ \boldsymbol{S} \end{pmatrix} = \begin{pmatrix} \tilde{\boldsymbol{A}}_1\boldsymbol{S} \\ \tilde{\boldsymbol{A}}_2\boldsymbol{S} \\ \vdots \\ \tilde{\boldsymbol{A}}_{\ell}\boldsymbol{S} \end{pmatrix}$$

then the statistical distance of $f(\boldsymbol{S})$ from the uniform over $\mathbb{Z}_q^{n\times k}$ is negligible ($2^{-\lambda}$) if $\mu \log(2d+1) \geq k\ell n + 2\lambda$. So if we set ℓ_{max} to be the maximum number of re-randomizations of the public key and $\mu \geq (k\ell_{\mathsf{max}}n + 2\lambda)/\log(2d+1)$ then we get class-hiding. The rest of the lattice parameters are set according to [Lyu12].

Unforgeability then comes directly from the unforgeability of [Lyu12] signatures and Accumulation Soundness is analogous to the Schnorr Signatures AKRS construction. Due to space limitations we postpone the detailed proofs for the full version.

Compiling to UCL and UCAL. As in the Schnorr signatures case, the overhead of bootstrapping a ring signature scheme to a UCAL one with the above AKRS is minimal. We further note that concrete costs of our AKRS (and thus the compilation to UCAL) can be optimized using follow-up optimizations on the Lyubashevsky signatures [DDLL13]. For UCL any general purpose NIZK for lattice relations can be used; our AKRS language is a basic lattice one.

6 Conclusions

In this paper, we have introduced Ring Signatures (RS) with User-Controlled Linkability (UCL) and User-Controlled Autonomous Linkability (UCAL). RS-UCL allows for both implicit and explicit linkability of signatures. Thus, signers can decide to make their signatures linkable either when issuing the signatures (by using the same scope in all signatures to be linked) or at a later time (by providing an explicit linking proof). We note that UCL was recently defined for group signatures. However, we argue that ring signatures are better suited for distributed applications, as no group manager is necessary. As such RS-UCL finds direct applicability in smart metering or smart mobility applications as argued in Sect. 1. Also, RS-UCL may be used in e-voting protocols where each election could use a different scope so that (i) double-voting in the same election round would be detected by implicit linkability, and (ii) voters can use the same (registered) key across election rounds.

RS-UCAL gives even more power to signers as they now can ensure unlinkability of their signatures, even if signatures use the same scope. Still, at a later time signers can prove linkability with an explicit linking proof.

We show how to upgrade *any* RS to RS-UCAL by means of a new cryptographic primitive that we have introduced in this paper and that we have labelled Anonymous Key Randomisable Signatures (AKRS). We have also shown how AKRS can be used to instantiate RS-UCL. We note that AKRS may be of independent interest and we have introduced two AKRS instantiations, one in prime-order groups and one based on lattices.

Acknowledgements. his work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under projects PICOCRYPT (grant agreement No. 101001283), and TERMINET (grant agreement No. 957406), by the Spanish Government under projects SCUM (ref. RTI2018-102043-B-I00), and RED2018-102321-T, by the Madrid Regional

Government under project BLOQUES (ref. S2018/TCS-4339). This work is also supported by a grant from Nomadic Labs and the Tezos foundation.

A Full Definitions for our AKRS Model

We here provide the full experiment for our AKRS existential unforgeability under chosen message attacks requirement.

```
Experiment :$\mathsf{Exp}^{\mathsf{euf-cma}}_{\mathsf{AKRS},\mathcal{A}}(1^\lambda)$
$Q \leftarrow \{\}, (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{pk}', \{t_1, \cdots, t_m\}, m, \sigma) \leftarrow \mathcal{A}^{\mathsf{OChgPK},\mathsf{OSign}}(\mathsf{pk})$
if $\mathsf{Vf}(\mathsf{pk}', m, \sigma) = 1 \wedge (m, \{t_1, \cdots, t_m\},) \notin Q$
 $\wedge\ \mathsf{pk}' = \mathsf{Accum}((t_1, \mathsf{ChgPK}(\mathsf{sk}, t_1)), \cdots, (t_m, \mathsf{ChgPK}(\mathsf{sk}, t_m)))$
    thenreturn $1$
else return $0$
```

```
$\mathsf{OChgPK}(t)$
$\mathsf{pk}' \leftarrow \mathsf{ChgPK}(\mathsf{sk}, t)$
return $\mathsf{pk}'$
```

```
$\mathsf{OSign}_Q(m, \mathsf{pk}', \{t_1, \cdots, t_m\})$
if $t \neq *$ then
   if $\mathsf{pk}' \neq \mathsf{Accum}((t_1, \mathsf{ChgPK}(\mathsf{sk}, t_1)), \cdots, (t_m, \mathsf{ChgPK}(\mathsf{sk}, t_m)))$
      thenreturn $\perp$
   $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{pk}', m)$
if $t = *$ then$\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{pk}, m)$
$Q \leftarrow Q \cup \{(m, t)\}$
return $\sigma$
```

B Full Definitions for our Ring Signature Models

We here provide the full definitions for ring signatures with user controlled (autonomous) linking that were omitted from the main body of the paper.

Signature Unforgeability for RS-UCAL. We next provide the full experiment for the signature unforgeability requirement in the RS-UCAL model.

```
Experiment :$\mathsf{Exp}^{\mathsf{sig\text{-}uf\text{-}al}}_{\mathsf{UCAL},\mathcal{A}}(1^\lambda, n, k)$
for $i = 1, \ldots, n$
   $(\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
for $i = 1, \ldots, k$   $\mathsf{ls}_i \leftarrow \mathsf{GenLinkSec}(1^\lambda)$
$\Sigma^* := (m^*, \mathsf{scp}^*, R, \sigma^*, \mathsf{nym}^*) \leftarrow \mathcal{A}^{\mathsf{OSign}^{\mathbf{ucl\text{-}r}},\mathsf{OLink},\mathsf{Corr},\mathsf{OLS}}(\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$
if $\mathsf{Vf}(\Sigma^*) = 0$ thenreturn $0$
if $R^* \subseteq \{\mathsf{vk}_i\}_{i \in [n] \setminus I}$   // none of the keys in $R^*$ were corrupted
    thenif $\forall i \in [n], (m^*, \mathsf{scp}^*, R^*, *, *) \notin \mathsf{SIG}[i, \cdot]$
       thenreturn $1$
else return $0$
```

Non-Frameability. We now provide the full experiments for both the signature non-frameability and link non-frameability requirements in the RS-UC AL model.

```
Experiment :$\mathsf{Exp}^{\mathsf{sign\text{-}frame}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k)$

$\textbf{for } i = 1, \dots, n \quad (\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
$\textbf{for } i = 1, \dots, k \quad \mathsf{ls}_i \leftarrow \mathsf{GenLinkSec}(1^\lambda)$
$\Sigma := (m, \mathsf{scp}, R, \sigma, \mathsf{nym}) \leftarrow \mathcal{A}^{\mathsf{OSign}^{\mathsf{ucl\text{-}r}}, \mathsf{OLink}, \mathsf{Corr}, \mathsf{OLS}}(\mathsf{vk}_1, \dots, \mathsf{vk}_n)$
$\textbf{return } 1 \quad \textbf{if} \quad :$
    $\mathsf{Vf}(\Sigma) = 1$ and
    $\exists i \in [n] : (m, \mathsf{scp}, R, \mathsf{nym}, \cdot) \notin \mathsf{SIG}[i] \wedge i \notin I \wedge (\cdot, \mathsf{scp}, \cdot, \cdot, \mathsf{nym}) \in \mathsf{SIG}[i]$
    $\exists i \in [k] : \Sigma \notin \mathsf{SIG}[\cdot, i] \wedge i \notin J \wedge (\cdot, \mathsf{scp}, \cdot, \cdot, \mathsf{nym}) \in \mathsf{SIG}[\cdot, i]$
$\textbf{else return } 0$
```

```
Experiment :$\mathsf{Exp}^{\mathsf{link\text{-}frame}}_{\mathsf{UCL},\mathcal{A}}(1^\lambda, n, k)$

$\textbf{for } i = 1, \dots, n \quad (\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KGen}(1^\lambda)$
$\textbf{for } i = 1, \dots, k \quad \mathsf{ls}_i \leftarrow \mathsf{GenLinkSec}(1^\lambda)$
$(\mathsf{lm}, \mathbf{\Sigma}, \pi_l) \leftarrow \mathcal{A}^{\mathsf{OSign}^{\mathsf{ucl\text{-}r}}, \mathsf{OLink}, \mathsf{Corr}, \mathsf{OLS}}(\mathsf{vk}_1, \dots, \mathsf{vk}_N)$
$\textbf{if } \mathsf{VerifyLink}(\mathsf{lm}, \mathbf{\Sigma}, \pi_l) = 0 \quad \textbf{return } 0$
Parse $\mathbf{\Sigma} = \{\Sigma_1, \cdots \Sigma_m\}$
$\forall j \in [m]$
    $\textbf{if } \exists i \in [n] : \Sigma_j \in \mathsf{SIG}[i] \textbf{ then} i_j \leftarrow i$
    $\textbf{if } \exists i \in [k] : \Sigma_j \in \mathsf{SIG}[\cdot, i] \textbf{ then} i_j \leftarrow i$
    $\textbf{elseif } \exists i \in [n] : \mathsf{Identify}(\mathsf{sk}_i, \mathsf{vk}_i, \Sigma_j) = 1 \textbf{ then} i_j \leftarrow i$
    $\textbf{elseif } \exists i \in [k] : \mathsf{Identify}(\mathsf{ls}_i, \Sigma_j) = 1 \textbf{ then} i_j \leftarrow i$
    $\textbf{else } i_j \leftarrow n + 1$
$\textbf{if } \exists j_1, j_2 \in [m] \quad i_{j_1} \neq i_{j_2} \quad \textbf{return } 1$
$\textbf{else return } 0$
```

References

[ACF21] Attema, T., Cramer, R., Fehr, S.: Compressing proofs of k-out-of-n partial knowledge. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12828, pp. 65–91. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84259-8_3

[ACHO13] Abe, M., Chow, S.S.M., Haralambiev, K., Ohkubo, M.: Double-trapdoor anonymous tags for traceable signatures. Int. J. Inf. Secur. **12**(1), 19–31 (2013)

[BCC04] Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: ACM CCS (2004)

[BCC+15] Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on DDH. In: Pernul, G., Ryan, P.Y.A., Weippl, E. (eds.) ESORICS 2015. LNCS, vol. 9326, pp. 243–265. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24174-6_13

[BCC+16] Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J.: Foundations of fully dynamic group signatures. In: ACNS (2016)

[BFG+13] Bernhard, D., Fuchsbauer, G., Ghadafi, E., Smart, N.P., Warinschi, B.: Anonymous attestation with user-controlled linkability. Int. J. Inf. Secur. **12**(3), 219–249 (2013)

[BHKS18] 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. LNCS, vol. 11273, pp. 405–434. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03329-3_14

[BMW03] Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_38

[BSZ05] Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: the case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30574-3_11

[CDL16a] Camenisch, J., Drijvers, M., Lehmann, A.: Anonymous attestation using the strong Diffie Hellman assumption revisited. In: Franz, M., Papadimitratos, P. (eds.) Trust 2016. LNCS, vol. 9824, pp. 1–20. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45572-3_1

[CDL16b] Camenisch, J., Drijvers, M., Lehmann, A.: Universally composable direct anonymous attestation. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016. LNCS, vol. 9615, pp. 234–264. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49387-8_10

[CS97] Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052252

[CvH91] Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_22

[DDLL13] Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 40–56. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_3

[DL21] Diaz, J., Lehmann, A.: Group signatures with user-controlled and sequential linkability. In: Garay, J.A. (ed.) PKC 2021. LNCS, vol. 12710, pp. 360–388. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75245-3_14

[FGL21] Fraser, A., Garms, L., Lehmann, A.: Selectively linkable group signatures—stronger security and preserved verifiability. In: Conti, M., Stevens, M., Krenn, S. (eds.) CANS 2021. LNCS, vol. 13099, pp. 200–221. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92548-2_11

[GK15] Groth, J., Kohlweiss, M.: One-out-of-many proofs: or how to leak a secret and spend a coin. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 253–280. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_9

[GL19] Garms, L., Lehmann, A.: Group signatures with selective linkability. In: PKC (2019)

[HILL99] Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)

[HLC+11] Hwang, J.Y., Lee, S., Chung, B.-H., Cho, H. S., Nyang, D.: Short group signatures with controllable linkability. In: Workshop on Lightweight Security & Privacy: (LightSec) (2011)

[HLC+13] Hwang, J.Y., Lee, S., Chung, B.-H., Cho, H.S., Nyang, D.H.: Group signatures with controllable linkability for dynamic membership. Inf. Sci. **222**, 761–778 (2013)

[KTY04] Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 571–589. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_34

[LJ14] Libert, B., Joye, M.: Group signatures with message-dependent opening in the standard model. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 286–306. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_15

[LMN16] Libert, B., Mouhartem, F., Nguyen, K.: A lattice-based group signature scheme with message-dependent opening. In: Manulis, M., Sadeghi, A.-R., Schneider, S. (eds.) ACNS 2016. LNCS, vol. 9696, pp. 137–155. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39555-5_8

[LPQ18] Libert, B., Peters, T., Qian, C.: Logarithmic-size ring signatures with tight security from the DDH assumption. In: Lopez, J., Zhou, J., Soriano, M. (eds.) ESORICS 2018. LNCS, vol. 11099, pp. 288–308. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98989-1_15

[LRR+19] Lai, R.W.F., Ronge, V., Ruffing, T., Schröder, D., Thyagarajan, S.A.K., Wang, J.: Omniring: scaling private payments without trusted setup. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, pp. 31–48 (2019)

[LWW04] Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 325–335. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27800-9_28

[Lyu12] Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_43

[OSEH13] Ohara, K., Sakai, Y., Emura, K., Hanaoka, G.: A group signature scheme with unbounded message-dependent opening. In: ASIA-CCS (2013)

[PS19] Park, S., Sealfon, A.: It wasn't me! In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11694, pp. 159–190. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8_6

[RST01] Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32

[SALY17] Sun, S.-F., Au, M.H., Liu, J.K., Yuen, T.H.: RingCT 2.0: a compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency Monero. In: Foley, S.N., Gollmann, D., Snekkenes, E. (eds.) ESORICS 2017. LNCS, vol. 10493, pp. 456–474. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66399-9_25

[SEH+12] Sakai, Y., Emura, K., Hanaoka, G., Kawai, Y., Matsuda, T., Omote, K.: Group signatures with message-dependent opening. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 270–294. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36334-4_18

[SSU14] Slamanig, D., Spreitzer, R., Unterluggauer, T.: Adding controllable linkability to pairing-based group signatures for free. In: Chow, S.S.M., Camenisch, J., Hui, L.C.K., Yiu, S.M. (eds.) ISC 2014. LNCS, vol. 8783, pp. 388–400. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13257-0_23

[YEL+21] Yuen, T.H., Esgin, M.F., Liu, J.K., Au, M.H., Ding, Z.: *DualRing*: generic construction of ring signatures with efficient instantiations. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 251–281. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_10

[YSL+20] Yuen, T.H., et al.: RingCT 3.0 for blockchain confidential transaction: shorter size and stronger security. In: Bonneau, J., Heninger, N. (eds.) FC 2020. LNCS, vol. 12059, pp. 464–483. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51280-4_25

[ZWC19] Zhang, T., Wu, H., Chow, S.S.M.: Structure-preserving certificateless encryption and its application. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 1–22. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_1