Efficient Sanitizable Signatures Without Random Oracles

Russell W.F. Lai[1], Tao Zhang[1], Sherman S.M. Chow[1(✉)], and Dominique Schröder[2]

[1] Department of Information Engineering, The Chinese University of Hong Kong, Sha Tin, N.T., Hong Kong
{wflai,zt112,sherman}@ie.cuhk.edu.hk
[2] Chair for Applied Cryptography, Friedrich-Alexander University Erlangen-Nürnberg, Erlangen and Nuremberg, Bavaria, Germany
schroeder@me.com

Abstract. Sanitizable signatures, introduced by Ateniese et al. (ESORICS '05), allow the signer to delegate the sanitization right of signed messages. The sanitizer can modify the message and update the signature accordingly, so that the sanitized part of the message is kept private. For stronger protection of sensitive information, it is desirable that no one can link sanitized message-signature pairs of the same document. This idea was formalized by Brzuska et al. (PKC '10) as unlinkability, which was followed up recently by Fleischhacker et al. (PKC '16). Unfortunately, these generic constructions of sanitizable signatures, unlinkable or not, are based on building blocks with specially crafted features which efficient (standard model) instantiations are absent. Basing on existing primitives or a conceptually simple primitive is more desirable.

In this work, we present two such generic constructions, leading to efficient instantiations in the standard model. The first one is based on rerandomizable tagging, a new primitive which may find independent interests. It captures the core accountability mechanism of sanitizable signatures. The second one is based on accountable ring signatures (CARDIS '04, ESORICS '15). As an intermediate result, we propose the first accountable ring signature scheme in the standard model.

1 Introduction

Regular signatures are non-malleable. It is infeasible to maul a valid message-signature pair (m, σ) into a modified pair (m', σ') that passes the verification. However, a controlled form of malleability can be desirable in many settings, such as research study on sanitized Internet traffic or anonymized medical data, commercial usages that replace advertisements in authenticated media streams, or updates of reliable routing information [2]. Sanitizable signatures, introduced by Ateniese *et al.* [2], support controlled malleability. The signer can specify parts of a (signed) message which a designated third party, called the sanitizer, is allowed to change and then adapt the signature accordingly. Brzuska *et al.* [10] formalized

I. Askoxylakis et al. (Eds.): ESORICS 2016, Part I, LNCS 9878, pp. 363–380, 2016.
DOI: 10.1007/978-3-319-45744-4_18

five security properties, including privacy which states that the sanitized part of the message cannot be recovered from a sanitized signature. A strictly stronger property, called unlinkability, was suggested one year later [11]. Unlinkability ensures that one cannot link sanitized message-signature pairs of the same document. It is particularly important in the motivating applications which sanitize data for privacy [2] as it prevents the attacker from combining information of several sanitized versions of a document for reconstructing (parts of) the original document. Such linkage is useful for de-anonymization.

Unlinkable sanitizable signatures was then constructed [11] from group signatures with an unusual property, that the keys of the signers can be computed independently even before seeing the keys of the group manager. In a typical application of group signature, the group is formed first and the signers join the group later. This order is even exploited for gaining efficiency in building group signature scheme via the notion of certified signatures [25]. In a very recent study of Fleischhacker *et al.* [23], in order to instantiate the generic construction of Brzuska *et al.* [11], they need to use an *inefficient* scheme based on the random oracle model (ROM) and generic group model (GGM) [24], or look into the details of the scheme [25] and perform the adaption accordingly to fit with the special requirement. This diminishes the benefits of a generic construction. Although the scheme [25] is proven in the standard model without random oracle, the proof requires the adversary to only perform group operations on the given elements (generic group model or GGM). No existing simple assumption supports the proof. Their study suggested that, to this date, no efficient group signature scheme *that has the required properties* is known, which also means that no efficient unlinkable sanitizable signature scheme is known. In response, they gave another generic construction from signatures with re-randomizable keys, which is very efficient when instantiated with Schnorr signature, yet with security argued with the ROM heuristics. Unfortunately, the re-randomizable keys property is also an unusual property, as showcased by the original authors [23] that two pairing-based short signature schemes cannot serve as a building block.

This leaves limited and unsatisfactory choices of schemes, (1) having a subset of the security properties [2,10], (2) relying on the ROM [23], or (3) secure without ROM, but building upon inefficient construction [11].

1.1 Our Contribution

Our main result is closing the research gap, presenting the first efficient (unlinkable) sanitizable signature schemes which are secure in the standard model. In fact, we propose two very different generic constructions which are both simple. Our study also gives several new results that are of independent interests.

Our first generic construction is based on *rerandomizable tagging* , a new notion which may find independent application. Indeed, it can be considered as a dual notion of double-trapdoor anonymous tag [1], a primitive proven to be useful for privacy-oriented authorship management mechanism. In particular, using it in a generic construction of traceable signature schemes allows the signer (or the group manager on behalf) to deny the authorship of a signature [1].

While both our tags and the public-keys expected by the signature scheme required in the previous generic construction [23] are "re-randomizable", we believe that our formulation captures the essential functionality to achieve accountability, for either creation or sanitization. This leads to our conceptually simple generic construction, in which the rerandomizable tagging scheme takes care of the accountability, a regular signature scheme for the signing functionality, a public-key encryption scheme for delegating signing power, and finally a pseudorandom function family for storing the randomness without maintaining local state. Using only basic primitives and our new rerandomizable tagging *without any zero-knowledge proof*, this construction is very efficient and achieves privacy, in the standard model and under only the relatively simpler static assumptions.

Our second generic construction, which achieves unlinkability, is based on *accountable ring signatures* [31]. In contrast to the existing generic construction from group signatures [11], where the latter is required to satisfy some special property, our construction relies on an existing notion which can be used as-is. One can immediately instantiate our construction by a recent scheme [7], which yields an efficient unlinkable sanitizable signature scheme in the ROM. As an extra feature, this generic construction naturally supports *multiple sanitizers* [16].

Aiming at constructing unlinkable sanitizable signatures in the standard model, we also construct the first accountable ring signature scheme in the standard model. The assumption required by this scheme is a q-type assumption due to the membership proof [8]. Our scheme inherits the constant signature size from non-accountable schemes in the literature [8]. Existing scheme [7] only relying on the (static) decisional Diffie-Hellman assumption requires a logarithmic size. Due to existing results [6,7], it also leads to a constant-size instantiation of a strong variant of fully dynamic group signatures, in which group manager not only can enroll, but also revoke group members.

1.2 Related Work

Ateniese *et al.* [2] informally describe the following properties of sanitizable signatures. *Unforgeability* says that signatures can only be created by honest signers and sanitizers. *Immutability* demands only designated parts of the message can be modified by the (malicious) sanitizer. *Transparency* ensures the indistinguishability of signatures computed by the signer and the sanitizer (or more precisely, they are indistinguishable to *public verifiers*, which means anyone other than the signer and the sanitizer themselves). *Accountability* means that neither the malicious signer nor the malicious sanitizer can deny authorship of the message. When need arises, the signer can generate a *proof of authorship*.

These requirements were formalized by Brzuska *et al.* [10]. Since then, many works formalize various other properties. Note that transparency ensures that any public verifier cannot even notice if the message has been sanitized. *Unlinkability*, introduced by Brzuska *et al.* [11], takes a step further in which a sanitized signature cannot be linked to its original version. This is crucial for privacy.

It is tricky to get a right balance of accountability and transparency. Canard *et al.* [17] addressed the lack of accountability in the seminal work [2], yet at the cost of transparency. On the other hand, unconditional transparency is often undesirable, which motivates the need of accountability. The original accountability notion [2,10] is interactive since it needs the participation of the signer. A non-interactive version was later proposed [12], which allows a third party to determine if a message originates from the signer or the sanitizer, without any help from the signer. Nevertheless, non-interactive accountability and transparency cannot be achieved simultaneously [23], so we focus on schemes that have (interactive) accountability and transparency.

Holding the sanitizer accountable is a measure after the fact. Another idea is to limit the allowable sanitization [15,28]. However, unlinkability in this setting is even more complicated. For instance, one may want to also conceal the sets of allowed modifications [13]. Yet, it appears to be difficult to construct such a scheme efficiently. Recently, Derler and Slamanig [22] suggested an intermediate notion (weaker than unlinkability but stronger than privacy) as a compromise for achieving efficient construction. We remark that Canard *et al.* [16] considered multiple signers and sanitizers, with construction based on group signatures.

Malleable signatures were considered in many variations, such as homomorphic signatures [18,27], which allows public evaluation of functions on more than one signed messages, or redactable signatures [9,27], which allows parts of the message to be removable. They aim to solve related but different problems, and are not directly applicable in our motivating scenarios as discussed [2,10,11,23].

Delegation of signing right is considered in proxy signatures [4]. Yet, the signatures produced by the proxy are often publicly distinguishable from signatures created by the designator, which violates the transparency property of sanitizable signatures. Recent advances such as (delegatable) functional signatures [3] associate the signing right with a policy specifying which messages can be signed, or even arbitrary functions to be applied on the key and the messages, such that the policy or the function remain hidden. These works show theoretical solutions, but are too slow for practical use.

2 Rerandomizable Tagging Schemes

In a high level, the core of a sanitizable signature is a cryptographic object which is computed by the signer with some secret information embedded, but can be rerandomized by the sanitizer many times in an indistinguishable way. In addition, when the sanitizer changes the object, it will no longer match with the embedded secret, indicating that the signature is sanitized.

To capture the above functionality, we introduce a new primitive called rerandomizable tagging. In a rerandomizable tagging scheme, the tag issuer generates a tag using its private key with respect to a user's public key. The user can then use its own private key to rerandomize the tag which looks indistinguishable from the one issued by the issuer. When necessary, however, the tag issuer can generate a proof to claim or deny the authorship of a (rerandomized) tag.

2.1 Definition of Rerandomizable Tagging Schemes

Definition 1. *A rerandomizable tagging scheme* $\mathcal{RT} = (\mathsf{TGen}_\mathtt{I}, \mathsf{TGen}_\mathtt{U}, \mathsf{Tag}, \mathsf{ReTag}, \mathsf{TVer}, \mathsf{TProv}, \mathsf{TJud})$ *consists of seven efficient algorithms:*

Key Generation. *The key generation algorithms for the issuer and the user respectively both create a pair of private and public key:* $(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$, $(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$.

Tagging. *The tagging algorithm takes as input a message* $m \in \{0,1\}^*$*, an issuer's private key* $\mathsf{sk}_\mathtt{I}$*, and a user public key* $\mathsf{pk}_\mathtt{U}$*. It outputs a tag* $\tau \leftarrow \mathsf{Tag}(m, \mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$.

Re-Tagging. *The re-tagging algorithm takes as input two messages* $m, m' \in \{0,1\}^*$*, the issuer's public key* $\mathsf{pk}_\mathtt{I}$*, a user's private key* $\mathsf{sk}_\mathtt{U}$*, and a tag* τ*. It outputs a new tag* $\tau' \leftarrow \mathsf{ReTag}(m, m', \mathsf{pk}_\mathtt{I}, \mathsf{sk}_\mathtt{U}, \tau)$.

Verification. *The verification algorithm takes as input a message* $m \in \{0,1\}^*$*, a tag* τ*, the issuer's public key* $\mathsf{pk}_\mathtt{I}$*, a user's public key* $\mathsf{pk}_\mathtt{U}$*, and outputs a bit* $b \leftarrow \mathsf{TVer}(m, \tau, \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$.

Proof. *The proof algorithm takes as input the issuer's private key* $\mathsf{sk}_\mathtt{I}$*, a message* $m \in \{0,1\}^*$*, a user's public key* $\mathsf{pk}_\mathtt{U}$*, and a tag* τ*. It outputs a proof* $\pi \leftarrow \mathsf{TProv}(\mathsf{sk}_\mathtt{I}, m, \mathsf{pk}_\mathtt{U}, \tau)$.

Judge. *The judge algorithm takes as input a message* $m \in \{0,1\}^*$*, issuer and user public keys* $\mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}$*, a tag* τ*, and a proof* π*. It outputs a decision* $d \in \{\mathtt{I}, \mathtt{U}\}$ *indicating whether the tag was created by the issuer or the user:* $d \leftarrow \mathsf{TJud}(m, \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}, \tau, \pi)$.

A rerandomizable tagging scheme is correct if, for all parameters $\lambda \in \mathbb{N}$, for all keys generated from $(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$ and $(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$, for all tags generated from $\tau \leftarrow \mathsf{Tag}(m, \mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$ and $\tau' \leftarrow \mathsf{ReTag}(m, m', \mathsf{pk}_\mathtt{I}, \mathsf{sk}_\mathtt{U}, \tau)$, it holds that $\mathsf{TVer}(m, \tau, \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}) = 1$ and $\mathsf{TVer}(m', \tau', \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}) = 1$. Furthermore, for all $\pi \leftarrow \mathsf{TProv}(\mathsf{sk}_\mathtt{I}, m, \mathsf{pk}_\mathtt{U}, \tau)$ and $\pi' \leftarrow \mathsf{TProv}(\mathsf{sk}_\mathtt{I}, m', \mathsf{pk}_\mathtt{U}, \tau')$, it holds that $\mathsf{TJud}(m, \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}, \tau, \pi) = \mathtt{I}$ and $\mathsf{TJud}(m', \mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U}, \tau', \pi') = \mathtt{U}$.

2.2 Security of Rerandomizable Tagging Schemes

Rerandomizable tagging schemes abstract the core properties of sanitizable signatures. Therefore, their security properties, namely, privacy, accountability, and transparency, follow the corresponding property of sanitizable signatures [10].

Privacy. This property says that the rerandomized tag should be hiding. Note that information leakage through the new message itself can never be prevented.

Definition 2 (Privacy). *A rerandomizable tagging scheme* $\mathcal{RT}$ *is* private *if for all PPT adversaries* $\mathcal{A}$ *the probability that the experiment* $\mathsf{TPrivacy}_\mathcal{A}^{\mathcal{RT}}(\lambda)$ *evaluates to* 1 *is negligibly close to* $\frac{1}{2}$ *(in* λ*), where*

Experiment $\mathsf{TPrivacy}^{\mathcal{RT}}_{\mathcal{A}}(\lambda)$
$(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$; $(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$; $b \leftarrow \{0,1\}$
$a \leftarrow \mathcal{A}^{\mathsf{Tag}(\cdot,\mathsf{sk}_\mathtt{I},\cdot),\mathsf{ReTag}(\cdot,\cdot,\cdot,\mathsf{sk}_\mathtt{U},\cdot),\mathsf{TProv}(\mathsf{sk}_\mathtt{I},\cdot,\cdot,\cdot),\mathsf{LoRReTag}(\cdot,\cdot,\mathsf{sk}_\mathtt{I},\mathsf{sk}_\mathtt{U},b)}(\mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$
where oracle $\mathsf{LoRReTag}(\cdot,\cdot,\mathsf{sk}_\mathtt{I},\mathsf{sk}_\mathtt{U},b)$, *on input* $((m_{j,0}, m'_j), (m_{j,1}, m'_j))$
computes $\tau_{j,b} \leftarrow \mathsf{Tag}(m_{j,b}, \mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$ *and returns*
$\tau'_{j,b} \leftarrow \mathsf{ReTag}(m_{j,b}, m'_j, \mathsf{pk}_\mathtt{I}, \mathsf{sk}_\mathtt{U}, \tau_{j,b})$
if $a = b$ *then output* 1, *else output* 0.

Accountability. This property demands that the origin of a (possibly rerandomized) tag should be undeniable. We distinguish between *issuer-accountability* and *user-accountability*. The former says that, if a tag has not been rerandomized, then a malicious issuer cannot make the judge accuse the user. In the issuer-accountability game, a malicious issuer $\mathcal{A}_{\mathsf{Tag}}$ gets a user public key $\mathsf{pk}_\mathtt{U}$ as input and has access to a re-tagging oracle, which takes as input tuples $(\mathsf{pk}_{\mathtt{I},i}, \tau_i)$ and returns τ'_i. Eventually, $\mathcal{A}_{\mathsf{Tag}}$ outputs a tuple $(\mathsf{pk}^*_\mathtt{I}, m^*, \tau^*, \pi^*)$ and wins the game if TJud accuses the user for the new key $\mathsf{pk}^*_\mathtt{I}$ with a valid tag τ^*.

Definition 3 (Issuer-Accountability). *A rerandomizable tagging scheme* $\mathcal{RT}$ *is* issuer-accountable *if for all PPT adversaries* $\mathcal{A}_{\mathsf{Tag}}$ *the probability that the experiment* $\mathsf{Iss\text{-}Acc}^{\mathcal{RT}}_{\mathcal{A}_{\mathsf{Tag}}}(\lambda)$ *outputs* 1 *is negligible (in* λ*), where*

Experiment $\mathsf{Iss\text{-}Acc}^{\mathcal{RT}}_{\mathcal{A}_{\mathsf{Tag}}}(\lambda)$
$(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$; $(\mathsf{pk}^*_\mathtt{I}, m^*, \tau^*, \pi^*) \leftarrow \mathcal{A}^{\mathsf{ReTag}(\cdot,\cdot,\mathsf{sk}_\mathtt{U},\cdot)}_{\mathsf{Tag}}(\mathsf{pk}_\mathtt{U})$
where $(\mathsf{pk}_{\mathtt{I},i}, m_i, m'_i, \tau_i)$ *and* τ'_i *denote the queries and answers to and from oracle* ReTag.
Output 1 if for all i *the following holds:*
$(\mathsf{pk}^*_\mathtt{I}, m^*) \neq (\mathsf{pk}_{\mathtt{I},i}, m'_i) \wedge \mathsf{TVer}(m^*, \tau^*, \mathsf{pk}^*_\mathtt{I}, \mathsf{pk}_\mathtt{U}) = 1$
$\wedge \mathsf{TJud}(m^*, \mathsf{pk}^*_\mathtt{I}, \mathsf{pk}_\mathtt{U}, \tau^*, \pi^*) \neq \mathtt{I}$
else output 0.

In the user-accountability game, $\mathcal{A}_{\mathsf{ReTag}}$ models a malicious user with access to Tag and TProv oracles. It succeeds if it outputs a message m^*, a tag τ^*, and a key $\mathsf{pk}^*_\mathtt{U}$ such that the output is different from $\mathsf{pk}_{\mathtt{U},i}$ previously queried to the Tag oracle. Moreover, it is required that the proof produced by the issuer via TProv still leads the judge to decide "$\mathtt{I}$", *i.e.*, the tag was created by the issuer.

Definition 4 (User-Accountability). *A rerandomizable tagging scheme* $\mathcal{RT}$ *is* user-accountable *if for all PPT adversaries* $\mathcal{A}_{\mathsf{ReTag}}$ *the probability that the experiment* $\mathsf{Usr\text{-}Acc}^{\mathcal{RT}}_{\mathcal{A}_{\mathsf{ReTag}}}(\lambda)$ *evaluates to* 1 *is negligible (in* λ*), where*

Experiment $\mathsf{Usr\text{-}Acc}^{\mathcal{RT}}_{\mathcal{A}_{\mathsf{ReTag}}}(\lambda)$
$(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$; $(m^*, \mathsf{pk}^*_\mathtt{U}, \tau^*) \leftarrow \mathcal{A}^{\mathsf{Tag}(\cdot,\mathsf{sk}_\mathtt{I},\cdot),\mathsf{TProv}(\mathsf{sk}_\mathtt{I},\cdot,\cdot,\cdot)}_{\mathsf{ReTag}}(\mathsf{pk}_\mathtt{I})$
where $(m_i, \mathsf{pk}_{\mathtt{U},i})$ *and* τ_i *denote the queries and answers of oracle* Tag.
$\pi \leftarrow \mathsf{TProv}(\mathsf{sk}_\mathtt{I}, m^*, \mathsf{pk}^*_\mathtt{U}, \tau^*)$
Output 1 if for all i *the following holds:*
$(\mathsf{pk}^*_\mathtt{U}, m^*) \neq (\mathsf{pk}_{\mathtt{U},i}, m_i) \wedge \mathsf{TVer}(m^*, \tau^*, \mathsf{pk}_\mathtt{I}, \mathsf{pk}^*_\mathtt{U}) = 1$
$\wedge \mathsf{TJud}(m^*, \mathsf{pk}_\mathtt{I}, \mathsf{pk}^*_\mathtt{U}, \tau^*, \pi) \neq \mathtt{U}$
else output 0.

Transparency. This property says that one cannot decide if a tag has been rerandomized or not. Formally, this is defined in a game where an adversary $\mathcal{A}$ has access to Tag, ReTag, and TProv oracles to create (rerandomized) tags and learn the proofs. In addition, $\mathcal{A}$ gets access to a $\mathsf{Tag/ReTag}_b(\cdot,\cdot)$ oracle with a secret random bit $b \in \{0,1\}$ embedded which, on input a messages m and m', behaves as follows:

- for $b = 0$ runs the tagging algorithm to create $\tau \leftarrow \mathsf{Tag}(m, \mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$, then runs the re-tagging algorithm $\tau' \leftarrow \mathsf{ReTag}(m, m', \mathsf{pk}_\mathtt{I}, \mathsf{sk}_\mathtt{U}, \tau)$ and returns the rerandomized tag τ',
- for $b = 1$ runs the tagging algorithm to create $\tau' \leftarrow \mathsf{Tag}(m', \mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$, then returns the tag τ'.

Adversary $\mathcal{A}$ eventually produces an output a as a guess for b. A rerandomizable tagging is *transparent* if for all efficient algorithms $\mathcal{A}$ the probability for a right guess $a = b$ in the above game is negligibly close to $\frac{1}{2}$. Below we also define a relaxed version called *proof-restricted transparency.*

Definition 5 ((Proof-Restricted) Transparency). *A rerandomizable tagging scheme $\mathcal{RT}$ is* proof-restrictedly transparent *if for all PPT adversaries $\mathcal{A}$ the probability that the experiment $\mathsf{Trans}^{\mathcal{RT}}_{\mathcal{A}}(\lambda)$ returns 1 is negligibly close to $\frac{1}{2}$.*

Experiment $\mathsf{Trans}^{\mathcal{RT}}_{\mathcal{A}}(\lambda)$
$(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$; $(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$; $b \leftarrow \{0,1\}$
$a \leftarrow \mathcal{A}^{\mathsf{Tag}(\cdot,\mathsf{sk}_\mathtt{I},\cdot),\mathsf{ReTag}(\cdot,\cdot,\mathsf{sk}_\mathtt{U},\cdot),\mathsf{TProv}(\mathsf{sk}_\mathtt{I},\cdot,\cdot,\cdot),\mathsf{Tag/ReTag}_b(\cdot,\cdot)}(\mathsf{pk}_\mathtt{I}, \mathsf{pk}_\mathtt{U})$
Output 1 if $\big(a = b \wedge M_{\mathsf{Tag/ReTag}} \cap M_{\mathsf{TProv}} = \emptyset\big)$ *else output* 0
where $M_{\mathsf{Tag/ReTag}}$ and M_{TProv} denote the sets of messages output from and queried to oracles $\mathsf{Tag/ReTag}$ *and* TProv *respectively.*

2.3 Construction of Rerandomizable Tagging Schemes

We describe a construction of rerandomizable tagging based on double-trapdoor chameleon hashing [19] and one-way functions. A double-trapdoor chameleon hash function is a chameleon hash function [29] for which there exists an efficient algorithm which takes as input a pair of collisions and outputs one of the trapdoors. Its formal definition can be found in the full version.

Informal Description. In our construction, the user public key is a public key of the double-trapdoor chameleon hashing scheme. A tag of a message m mainly consists of the randomness ρ used in computing a chameleon hash value of the message m. The pair (m, ρ) implicitly fixes the hash value μ. By the collision resistance of the chameleon hash, it is infeasible for the issuer to find another pair (m', ρ') which hashes to the same value μ, while the user can sample as many collisions as it wants, on arbitrary messages m'. Thus, to rerandomize a tag for a message m into a tag for another message m', the user simply replace ρ by ρ' such that the pairs (m, ρ) and (m', ρ') hash to the same value μ. To prove the authorship of this tag later, the issuer first obtains a random seed r

$\mathsf{TGen}_{\mathtt{I}}(1^\lambda)$

$K \leftarrow \{0,1\}^\lambda$
$(\mathsf{sk}_\Sigma, \mathsf{pk}_\Sigma) \leftarrow \mathsf{SGen}(1^\lambda)$
$\mathsf{sk}_{\mathtt{I}} := (K, \mathsf{sk}_\Sigma)$
$\mathsf{pk}_{\mathtt{I}} := \mathsf{pk}_\Sigma$
return $(\mathsf{sk}_{\mathtt{I}}, \mathsf{pk}_{\mathtt{I}})$

$\mathsf{TGen}_{\mathtt{U}}(1^\lambda)$

$(\mathsf{sk}_{\mathcal{H},0}, \mathsf{sk}_{\mathcal{H},1}, \mathsf{pk}_{\mathcal{H}}) \leftarrow \mathsf{CGen}(1^\lambda)$
$\mathsf{sk}_{\mathtt{U}} := \mathsf{sk}_{\mathcal{H},0}$
$\mathsf{pk}_{\mathtt{U}} := \mathsf{pk}_{\mathcal{H}}$
return $(\mathsf{sk}_{\mathtt{U}}, \mathsf{pk}_{\mathtt{U}})$

$\mathsf{ReTag}(m, m', \mathsf{pk}_{\mathtt{I}}, \mathsf{sk}_{\mathtt{U}}, \tau)$

$\rho' \leftarrow \mathsf{CInv}(\mathsf{sk}_{\mathcal{H},0}, m, \rho, m')$
$\tau' := (\rho', q, \sigma)$
return τ'

$\mathsf{Tag}(m, \mathsf{sk}_{\mathtt{I}}, \mathsf{pk}_{\mathtt{U}})$

$q \leftarrow \{0,1\}^\lambda$
$r \leftarrow F(K, q)$
$\rho \leftarrow g(r)$
$\mu \leftarrow \mathsf{CEval}(\mathsf{pk}_{\mathcal{H}}, m; \rho)$
$\eta := (\mathsf{pk}_{\mathtt{U}}, q, \mu)$
$\sigma \leftarrow \mathsf{SSig}(\mathsf{sk}_\Sigma, \eta)$
$\tau := (\rho, q, \sigma)$
return τ

$\mathsf{Ver}(m, \tau, \mathsf{pk}_{\mathtt{I}}, \mathsf{pk}_{\mathtt{U}})$

$\mu \leftarrow \mathsf{CEval}(\mathsf{pk}_{\mathcal{H}}, m; \rho)$
$\eta := (\mathsf{pk}_{\mathtt{U}}, q, \mu)$
$b \leftarrow \mathsf{SVer}(\eta, \sigma, \mathsf{pk}_\Sigma)$
return b

$\mathsf{TProv}(\mathsf{sk}_{\mathtt{I}}, m, \mathsf{pk}_{\mathtt{U}}, \tau)$

$r \leftarrow F(K, q)$
$\pi := (m, r)$
return π

$\mathsf{TJud}(m, \mathsf{pk}_{\mathtt{I}}, \mathsf{pk}_{\mathtt{U}}, \tau, \pi)$

$\rho' \leftarrow g(r')$
$\mu \leftarrow \mathsf{CEval}(\mathsf{pk}_{\mathcal{H}}, m; \rho)$
$\mu' \leftarrow \mathsf{CEval}(\mathsf{pk}_{\mathcal{H}}, m'; \rho')$
if $\mu = \mu' \wedge \rho \neq \rho'$ **then**
  return $d = \mathtt{U}$
else
  return $d = \mathtt{I}$
endif

Fig. 1. Our rerandomizable tagging scheme

by evaluating a pseudorandom function on a random input q, then applies a pseudorandom generator on r and uses it as the randomness ρ for the chameleon hash. It signs the random input q with the randomness ρ. In the case of dispute, the issuer can recover q and hence r from the signature, and uses r as the proof of (non-)authorship.

Formal Description. Let $F : \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^\lambda$ be a pseudorandom function. Let $g : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a pseudorandom generator. Let $\mathcal{H} = (\mathsf{CGen}, \mathsf{TCGen}, \mathsf{CEval}, \mathsf{CInv})$ be a double-trapdoor chameleon hash which hashes messages $m \in \{0,1\}^*$ with randomness $\rho \in \{0,1\}^{2\lambda}$. Let $\Sigma = (\mathsf{SGen}, \mathsf{SSig}, \mathsf{SVer})$ be a signature scheme. We construct a rerandomizable tagging scheme $\mathcal{RT}$ as shown in Fig. 1. The correctness of $\mathcal{RT}$ follows those of $\mathcal{H}$ and Σ.

Theorem 1. *If $\mathcal{H}$ has uniform output distribution, then $\mathcal{RT}$ is private. If one-way function exists, then $\mathcal{RT}$ is user-accountable. If $\mathcal{H}$ is collision-resistant, then $\mathcal{RT}$ is issuer-accountable. If one-way function exists and $\mathcal{H}$ has uniform output distribution, then $\mathcal{RT}$ is proof-restrictedly transparent.*

Due to space constraint, detailed proofs can be found in the full version.

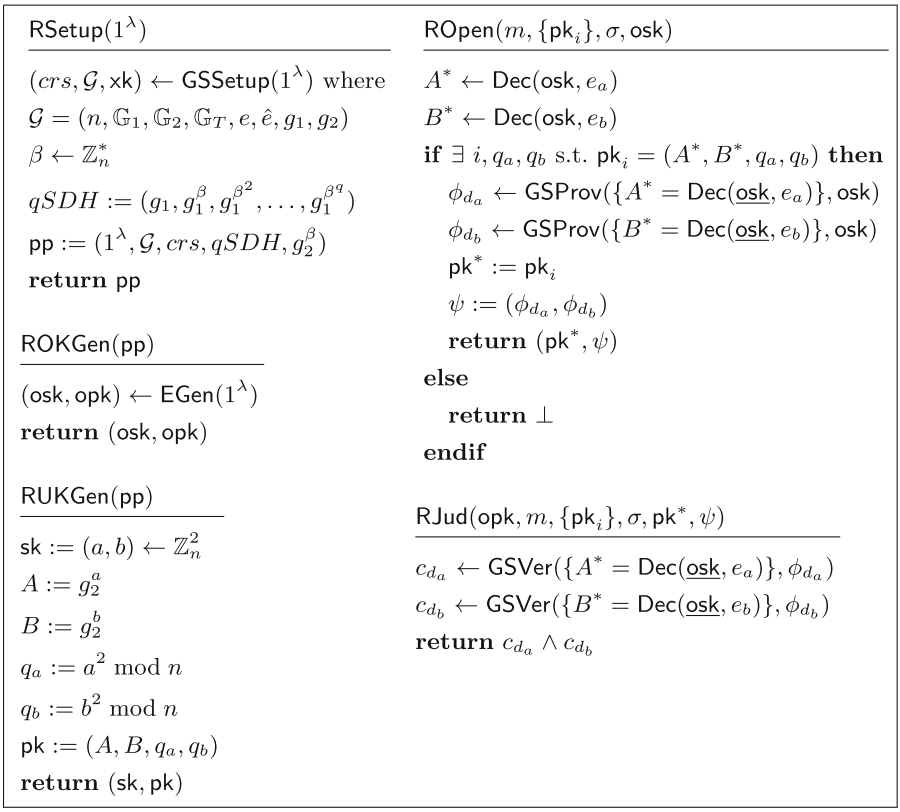

Fig. 2. Our accountable ring signature scheme - Part I

3 Accountable Ring Signatures

Accountable ring signatures, introduced by Xu and Yung [31] and recently formalized by Bootle *et al.* [7], allows both spontaneous group formulation as ring signatures and designated opening of signer identity as group signatures. Bootle *et al.* [7] gave a generic construction and an efficient instantiation in the random oracle model. We follow the the definitions of Bootle *et al.* [7], which can be found in the full version.

We adopt the ring signature scheme of Bose *et al.* [8] (referred to as BDR hereinafter), which in turn uses the Boneh-Boyen signature scheme [5] for signing hash values output by a collision-resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_n$. We transform BDR into an accountable ring signature scheme $\mathcal{RS}$, described in Figs. 2 and 3, by using a structure-preserving encryption scheme $\mathcal{SPE} = (\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec})$ of Camenisch *et al.* [14] which is secure against chosen-ciphertext attack (CCA). We use a collision-resistant hash function $H_2 : \mathbb{Z}_n \rightarrow \mathbb{G}_1$ to create a label for $\mathcal{SPE}$. Roughly, we encrypt the public key and prove using Groth-Sahai proof system [26] $\mathcal{GS} = (\mathsf{GSSetup}, \mathsf{GSProv}, \mathsf{GSVer})$

$\mathsf{RSig}(\mathsf{opk}, m, (\mathsf{pk}_1 = (q_{1,a}, q_{1,b}), \ldots, \mathsf{pk}, \ldots, \mathsf{pk}_k = (q_{k,a}, q_{k,b})), \mathsf{sk})$

$m' \leftarrow H(m||\{\mathsf{pk}_i\}); \rho \leftarrow \mathbb{Z}_n \setminus \{\frac{-a+m'}{b}\}; \Delta \leftarrow g_1^{\frac{1}{a+\rho b+m'}}$

$\mathcal{R}_a := (q_{1,a}, \ldots, q_{k,a}); \mathcal{R}_b := (q_{1,b}, \ldots, q_{k,b})$

$W_a \leftarrow \mathsf{MemWit}(\mathsf{pp}, q_a, \mathcal{R}_a); W_b \leftarrow \mathsf{MemWit}(\mathsf{pp}, q_b, \mathcal{R}_b)$

$\phi_{mem_a} \leftarrow \mathsf{MemProv}(\mathsf{pp}, \mathcal{R}_a, W_a); \phi_{mem_b} \leftarrow \mathsf{MemProv}(\mathsf{pp}, \mathcal{R}_b, W_b)$

$\phi_{q_a} \leftarrow \mathsf{GSProv}(\{\underline{q_a} = \underline{a}^2\}, (q_a, a)); \phi_{q_b} \leftarrow \mathsf{GSProv}(\{\underline{q_b} = \underline{b}^2\}, (q_b, b))$

$\phi_{\mathsf{pk}_a} \leftarrow \mathsf{GSProv}(\{\underline{A} = g_2^{\underline{a}}\}, (A, a)); \phi_{\mathsf{pk}_b} \leftarrow \mathsf{GSProv}(\{\underline{B} = g_2^{\underline{b}}\}, (B, b))$

$e_a \leftarrow \mathsf{Enc}(\mathsf{opk}, H_2(m'), A; r_a); e_b \leftarrow \mathsf{Enc}(\mathsf{opk}, H_2(m'), B; r_b)$

$\phi_{e_a} \leftarrow \mathsf{GSProv}(\{e_a = \mathsf{Enc}(\mathsf{opk}, H_2(m'), \underline{A}; \underline{r_a})\}, (A, r_a))$

$\phi_{e_b} \leftarrow \mathsf{GSProv}(\{e_b = \mathsf{Enc}(\mathsf{opk}, H_2(m'), \underline{B}; \underline{r_b})\}, (B, r_b))$

$\phi_{sig} \leftarrow \mathsf{GSProv}(\{\underline{B}^\rho = \underline{B'} \wedge e(\underline{\Delta}, \underline{A})e(\underline{\Delta}, \underline{B'})e(\underline{\Delta}, g_2^{m'}) = e(g_1, g_2)\}, (\Delta, A, B, B'))$

return $\sigma := (\rho, e_a, e_b, \phi_{mem_a}, \phi_{mem_b}, \phi_{sig}, \phi_{q_a}, \phi_{q_b}, \phi_{\mathsf{pk}_a}, \phi_{\mathsf{pk}_b}, \phi_{e_a}, \phi_{e_b})$

$\mathsf{RVer}(\mathsf{opk}, m, \{\mathsf{pk}_i = (q_{i,a}, q_{i,b})\}_{i=1}^k, \sigma)$

$m' \leftarrow H(m||\{\mathsf{pk}_i\}); \mathcal{R}_a := (q_{1,a}, \ldots, q_{k,a}); \mathcal{R}_b := (q_{1,b}, \ldots, q_{k,b})$

$c_{mem_a} \leftarrow \mathsf{MemVer}(\mathsf{pp}, \mathcal{R}_a, \phi_{mem_a}); c_{mem_b} \leftarrow \mathsf{MemVer}(\mathsf{pp}, \mathcal{R}_b, \phi_{mem_b})$

$c_{q_a} \leftarrow \mathsf{GSVer}(\{\underline{q_a} = \underline{a}^2\}, \phi_{q_a}); c_{q_b} \leftarrow \mathsf{GSVer}(\{\underline{q_b} = \underline{b}^2\}, \phi_{q_b})$

$c_{\mathsf{pk}_A} \leftarrow \mathsf{GSVer}(\{\underline{A} = g_2^{\underline{a}}\}, \phi_{\mathsf{pk}_A}); c_{\mathsf{pk}_B} \leftarrow \mathsf{GSVer}(\{\underline{B} = g_2^{\underline{b}}\}, \phi_{\mathsf{pk}_B})$

$c_{e_a} \leftarrow \mathsf{GSVer}(\{e_a = \mathsf{Enc}(\mathsf{opk}, H_2(m'), \underline{A}; \underline{r_a})\}, \phi_{e_a})$

$c_{e_b} \leftarrow \mathsf{GSVer}(\{e_b = \mathsf{Enc}(\mathsf{opk}, H_2(m'), \underline{B}; \underline{r_b})\}, \phi_{e_b})$

$c_{sig} \leftarrow \mathsf{GSVer}(\{\underline{B}^\rho = \underline{B'} \wedge e(\underline{\Delta}, \underline{A})e(\underline{\Delta}, \underline{B'})e(\underline{\Delta}, g_2^{m'}) = e(g_1, g_2)\}, \phi_{sig})$

return $(c_{mem_a} \wedge c_{mem_b} \wedge c_{sig} \wedge c_{q_a} \wedge c_{q_b} \wedge c_{\mathsf{pk}_A} \wedge c_{\mathsf{pk}_B} \wedge c_{e_a} \wedge c_{e_b})$

Fig. 3. Our accountable ring signature scheme - Part II

that the encrypted key matches with the one for verifying the BDR signature. A tracing authority holding the decryption key can identify the real signer. BDR ring signature requires composite order group, so our accountable ring signature is also constructed in a composite order setting. Our scheme inherits the nice features of BDR, including constant signature size and security without random oracles. We underline the witness components in the statement to be proven by Groth-Sahai proof system. The details of $\mathcal{SPE}$ and Groth-Sahai proof system can be found in the full version.

Analysis. The correctness of $\mathcal{RS}$ follows those of BDR ring signatures and the proof on $\mathcal{SPE}$ ciphertexts. The efficiency of $\mathcal{RS}$ depends on the instantiation of the Groth-Sahai proof. Instantiating $\mathcal{SPE}$ and our accountable ring signature

scheme with a composite order group, and the Groth-Sahai proof system with the symmetric external Diffie-Hellman (SXDH) assumption [26], there are 121 multiplications, 102 exponentiations (including the commitments for the proofs), and 10 pairings in the signing algorithm $\mathsf{RSig}()$.

Theorem 2. *If $\mathcal{GS}$ is sound and the underlying scheme [8] is unforgeable, $\mathcal{RS}$ is unforgeable. If $\mathcal{SPE}$ is CCA-secure, and $\mathcal{GS}$ is hiding, $\mathcal{RS}$ is CCA-anonymous under full key exposure. If $\mathcal{GS}$ is sound, $\mathcal{RS}$ is traceable and has tracing soundness.*

Due to space constraint, detailed proofs can be found in the full version.

4 Constructions of Sanitizable Signatures

Syntax. Sanitizable signature schemes allow the delegation of signing capabilities to a sanitizer. These capabilities are realized by letting the signer "attach" a description of the admissible modifications for a particular message and sanitizer. The sanitizers may then change the message according to some modification and update the signature. More formally, the signer uses its private key $\mathsf{sk_S}$ to sign a message m and the description of the admissible modifications α for some sanitizer $\mathsf{pk_Z}$. The sanitizer, having a matching private key $\mathsf{sk_Z}$, can update the message according to some modification δ and compute a new signature using $\mathsf{sk_Z}$. In case of a dispute about the origin of a message-signature pair, the signer can compute a proof π (using an algorithm Prov) from previously signed messages which proves that a signature has been created by the sanitizer. The verification of this proof is done by an algorithm Jud (that only decides the origin of a valid message-signature pair in question; for invalid pairs such decisions are in general impossible). We mostly follow the existing syntax [10,11] except that our key generation algorithms take as input a public parameter generated by a setup algorithm. For the formal syntax and security definitions of sanitizable signatures, readers can refer to the full version.

Sanitizable signatures should satisfy immutability, sanitizer- and signer-accountability, and proof-restricted transparency. In addition, they should satisfy privacy or, even stronger, unlinkability. It is known that full transparency or unlinkability both imply privacy separately [10,11], while proof-restricted transparency only implies a proof-restricted version of privacy [11].

To the best of our knowledge, there is no efficient instantiation of sanitizable signatures satisfying either privacy or unlinkability, and all other security properties simultaneously, without using random oracles. We thus fill this gap by describing two constructions. The first is more efficient while satisfying privacy based on the rerandomizable tagging. The second one uses the accountable ring signature scheme and can achieve unlinkability.

4.1 Privacy from Rerandomizable Tagging Scheme

Informal Description. Our first construction relies heavily on the rerandomizable tagging scheme (Sect. 2) which captures the accountability properties of sanitizable signatures. We complement it with pseudorandom functions, signatures,

$\underline{\mathsf{Setup}(1^\lambda)}$

$\mathsf{pp} = 1^\lambda$

return pp

$\underline{\mathsf{Prov}(\mathsf{sk}_\mathsf{S}, m, \sigma, \mathsf{pk}_\mathsf{Z})}$

$r_e \leftarrow F(K_e, q_e)$

$\pi := r_e$

return π

$\underline{\mathsf{KGen}_\mathsf{S}(\mathsf{pp})}$

$K_e \leftarrow \{0,1\}^\lambda$

$(\mathsf{sk}_\mathsf{f}, \mathsf{pk}_\mathsf{f}) \leftarrow \mathsf{SGen}(1^\lambda)$

$(\mathsf{sk}_\mathtt{I}, \mathsf{pk}_\mathtt{I}) \leftarrow \mathsf{TGen}_\mathtt{I}(1^\lambda)$

$\mathsf{sk}_\mathsf{S} = (K_e, \mathsf{sk}_\mathsf{f}, \mathsf{sk}_\mathtt{I})$

$\mathsf{pk}_\mathsf{S} = (\mathsf{pk}_\mathsf{f}, \mathsf{pk}_\mathtt{I})$

return $(\mathsf{sk}_\mathsf{S}, \mathsf{pk}_\mathsf{S})$

$\underline{\mathsf{KGen}_\mathsf{Z}(\mathsf{pp})}$

$(\mathsf{sk}_e, \mathsf{pk}_e) \leftarrow \mathsf{EGen}(1^\lambda)$

$(\mathsf{sk}_\mathtt{U}, \mathsf{pk}_\mathtt{U}) \leftarrow \mathsf{TGen}_\mathtt{U}(1^\lambda)$

$\mathsf{sk}_\mathsf{Z} = (\mathsf{sk}_e, \mathsf{sk}_\mathtt{U})$

$\mathsf{pk}_\mathsf{Z} = (\mathsf{pk}_e, \mathsf{pk}_\mathtt{U})$

return $(\mathsf{sk}_\mathsf{Z}, \mathsf{pk}_\mathsf{Z})$

Fig. 4. Our first sanitizable signature scheme - Part I

and extractable public key encryption to provide the functionality of delegating signing power of the signer to the sanitizers. The details of these primitives can be found in the full version.

To sign, the signer computes a tag τ of the message m using the rerandomizable tagging scheme. It generates a fresh key pair $(\hat{\mathsf{sk}}, \hat{\mathsf{pk}})$ of a signature scheme, and uses this newly generated $\hat{\mathsf{sk}}$ to sign m. The key pair $(\hat{\mathsf{sk}}, \hat{\mathsf{pk}})$ can be interpreted as the binding factor of the entire sanitizing chain. We assume there exists an efficient algorithm to check whether $\hat{\mathsf{sk}}$ and $\hat{\mathsf{pk}}$ forms a valid signing and verification key pair, denoted by $\hat{\mathsf{sk}} \sim \hat{\mathsf{pk}}$. To delegate the signing power to the sanitizer, the signer simply encrypts the fresh private key $\hat{\mathsf{sk}}$, together with some other identifying information, to the sanitizer. Finally, the signer uses its long term private key sk_S to sign the fixed part of the message $f_\alpha(m)$, the sanitizer's public key pk_Z, and the fresh public key $\hat{\mathsf{pk}}$, along with other information. This binds the sanitizer to the sanitizing chain identified by $\hat{\mathsf{pk}}$. The signature thus consists mainly of a signature σ_f of the fixed part, a signature $\hat{\sigma}$ signed by $\hat{\mathsf{sk}}$, the tag τ, the fresh public key $\hat{\mathsf{pk}}$, and the ciphertext c.

To sanitize, the sanitizer decrypts c to retrieve $\hat{\mathsf{sk}}$, and rerandomizes the tag with respect to the new message $m' = \delta(m)$ using the rerandomizable tagging scheme. It then uses $\hat{\mathsf{sk}}$ to sign m' and outputs it.

To extend the accountability of the rerandomizable tagging scheme to the sanitizable signature scheme, we use a similar technique as in the construction of the former: The signer generates the ciphertext using pseudorandomness generated from a random input, which is included in the sanitizable signature. The signer can later recover the pseudorandomness used for encryption by applying the pseudorandom function on this input. This pseudorandomness allows the judge to extract the identifying information from the ciphertext and verifies the authorship of the signature. The extractability of the encryption scheme relieves the signer of using any zero-knowledge proof in the proof algorithm.

Formal Description. Let $F : \{0,1\}^\lambda \times \{0,1\}^* \rightarrow \{0,1\}^\lambda$ be a pseudorandom function, $\Sigma = (\mathsf{SGen}, \mathsf{SSig}, \mathsf{SVer})$ be a digital signature scheme,

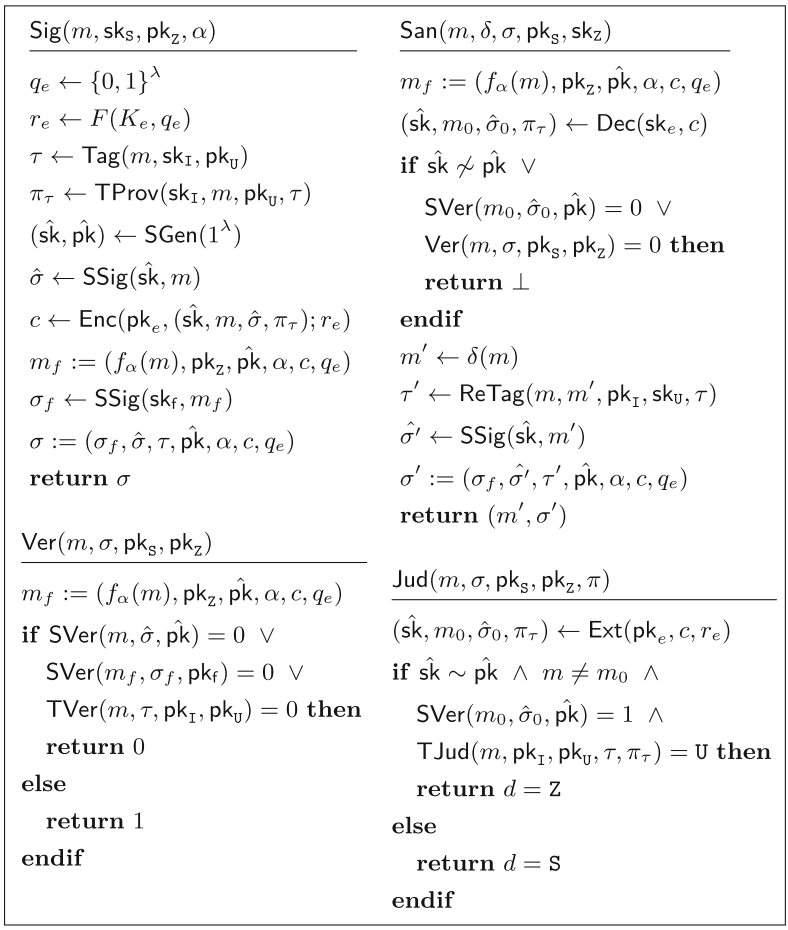

Fig. 5. Our first sanitizable signature scheme - Part II

$\mathcal{E} = (\mathsf{EGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Ext})$ be an extractable public key encryption scheme, and $\mathcal{RT} = (\mathsf{TGen_I}, \mathsf{TGen_U}, \mathsf{Tag}, \mathsf{ReTag}, \mathsf{TProv}, \mathsf{TJud})$ be a rerandomizable tagging scheme (Sect. 2). Figures 4 and 5 describe our first sanitizable signature scheme $\mathcal{SS}_1$. Its correctness follows directly from those of Σ, $\mathcal{E}$, and $\mathcal{RT}$.

Theorem 3. *If $\mathcal{RT}$ is private, then $\mathcal{SS}_1$ is private. If Σ is EUF-CMA secure, then $\mathcal{SS}_1$ is immutable. If Σ is EUF-CMA secure, $\mathcal{E}$ is correct, and $\mathcal{RT}$ is user-accountable, then $\mathcal{SS}_1$ is sanitizer-accountable. If $\mathcal{RT}$ is issuer-accountable, then $\mathcal{SS}_1$ is signer-accountable. If $\mathcal{RT}$ is proof-restrictedly transparent, then $\mathcal{SS}_1$ is proof-restrictedly transparent.*

Due to space constraint, detailed proofs can be found in the full version.

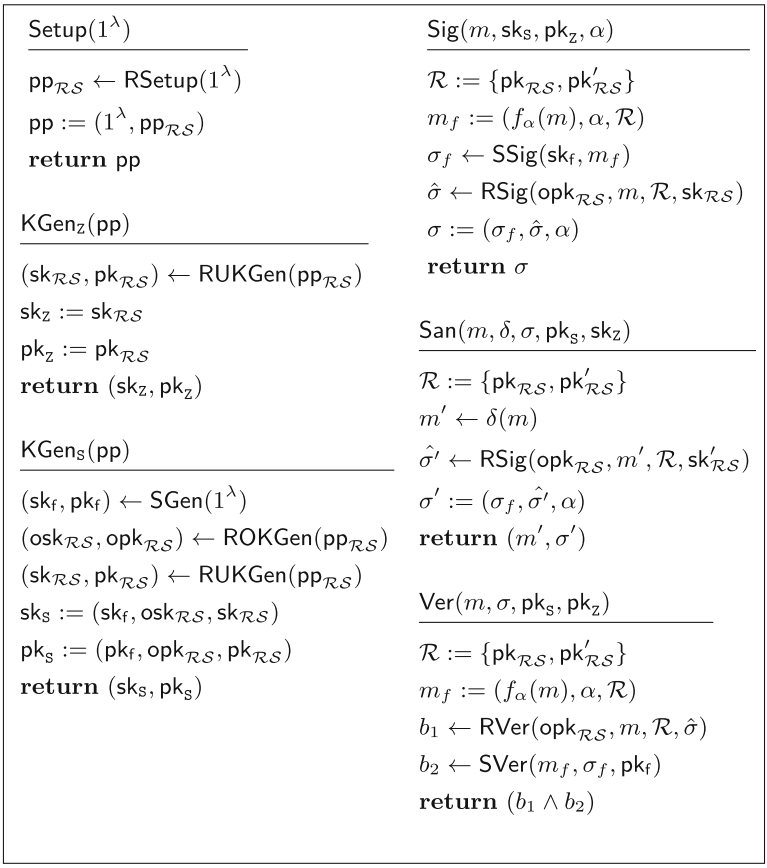

Fig. 6. Our second sanitizable signature scheme - Part I

4.2 Unlinkability from Accountable Ring Signatures

Our second construction is similar to the construction by Brzuska *et al.* [11] based on group signatures, except that we replace the special group signatures with accountable ring signatures reviewed in Sect. 3. This change has two interesting effects. First, the construction of sanitizable signatures becomes simpler: The signer does not need to create a new group for each sanitizable signature, which also eliminates the use of pseudorandom functions to generate the group [11]. Second, in contrast to the special group signatures, which the instantiations (with or without random oracle heuristics) are not efficient [23], our accountable ring signatures scheme in Sect. 3 is efficient and is secure without random oracles, though it requires composite order group.

Another route leading to our discovery is the observation that the fully dynamic group signatures constructed from accountable ring signatures [6,7] features the property that the user key generation does not depend on the group

$\mathsf{Prov}(\mathsf{sk_S}, m, \sigma, \mathsf{pk_Z})$

$\mathcal{R} := \{\mathsf{pk}_{\mathcal{RS}}, \mathsf{pk}'_{\mathcal{RS}}\}$
$(\mathsf{pk}^*_{\mathcal{RS}}, \psi) \leftarrow \mathsf{ROpen}(m, \mathcal{R}, \hat{\sigma}, \mathsf{osk}_{\mathcal{RS}})$
$\pi := (\mathsf{pk}^*_{\mathcal{RS}}, \psi)$
return π

$\mathsf{Jud}(m, \sigma, \mathsf{pk_S}, \mathsf{pk_Z}, \pi)$

$\mathcal{R} := \{\mathsf{pk}_{\mathcal{RS}}, \mathsf{pk}'_{\mathcal{RS}}\}$
if $\mathsf{RJud}(\mathsf{opk}_{\mathcal{RS}}, m, \mathcal{R}, \hat{\sigma}, \mathsf{pk}^*_{\mathcal{RS}}, \psi) = 1$
$\quad \wedge \ \mathsf{pk}^*_{\mathcal{RS}} = \mathsf{pk}'_{\mathcal{RS}}$ **then**
return $d := \mathtt{Z}$
else
return $d := \mathtt{S}$
endif

Fig. 7. Our second sanitizable signature scheme - Part II

key pair, which is the property required in the sanitizable signatures construction by Brzuska *et al.* [11].

Informal Description. We proceed directly to the signing and sanitizing procedures. To issue a signature, the signer forms a ring consisting of itself and the sanitizer, and ring-signs the message. It binds the sanitizer to this sanitizing chain by signing the fixed part of the message together with the sanitizer's public key using its private key. Sanitizing becomes computing a new accountable ring signature on the modified message.

Formal Description. Let $\mathcal{RS} = (\mathsf{RSetup}, \mathsf{ROKGen}, \mathsf{RUKGen}, \mathsf{RSig}, \mathsf{RVer}, \mathsf{ROpen}, \mathsf{RJud})$ be an accountable ring signature scheme (Sect. 3), and $\Sigma = (\mathsf{SGen}, \mathsf{SSig}, \mathsf{SVer})$ be a digital signature scheme. Our unlinkable sanitizable signature scheme $\mathcal{SS}_2$ is constructed in Figs. 6 and 7. The correctness of $\mathcal{SS}_2$ follows those of $\mathcal{RS}$ and Σ.

Multiple Sanitizers. Ring signatures support rings containing more than two members, so we can extend $\mathcal{SS}_2$ easily to support more sanitizers: The signer can sign the public keys of a ring of multiple sanitizers when issuing a sanitizable signature. This grants partial signing power to each the sanitizers (possibly corresponding to different admissible modifications). Furthermore, since our accountable ring signatures have constant signature size with respect to the number of users in the ring, the multiple sanitizer scheme also features constant signature size with respect to the number of sanitizers.

Theorem 4. *If Σ is EUF-CMA secure, then $\mathcal{SS}_2$ is immutable and unlinkable. If $\mathcal{RS}$ is traceable and satisfies tracing soundness, then $\mathcal{SS}_2$ is sanitizer accountable. If $\mathcal{RS}$ is traceable and satisfies tracing soundness, then $\mathcal{SS}_2$ is signer accountable. If $\mathcal{RS}$ is anonymous, then $\mathcal{SS}_2$ is proof-restrictedly transparent.*

Due to space constraint, detailed proofs can be found in the full version.

Table 1. Comparison of different sanitizable signature schemes

	$\mathcal{SS}_1$	$\mathcal{SS}_2$	[23]	[11] using [25]	[11] using [24]
Security	Privacy	Unlinkability	Unlinkability	Unlinkability	Unlinkability
Model	Standard	Standard	ROM	Standard	ROM
Assumption	Static	q-type	Static	GGM	GGM
$\mathsf{KGen_S}$	4E+2P	32E+1P	7E	1E	1E
$\mathsf{KGen_Z}$	7E	2E	1E	1E	4E
Sig	15E+1P	103E+10P	15E	194E+2P	2813E
San	4E+12P	102E+10P	14E	186E+1P	2814E
Ver	2E+9P	2E+148P	17E	207E+62P	2011E
Prov	0E+0P	126E+152P	23E	14E+1P	18E
Jud	4E+3P	152P	6E	1E+2P	2E
$\mathsf{pk_S}$, $\mathsf{sk_S}$	$10 + 2\|m\|, 3$	$26, 25$	$7, 14$	$1, 1$	$1, 1$
$\mathsf{pk_Z}$, $\mathsf{sk_Z}$	$5, 7$	$4, 2$	$1, 1$	$1, 1$	$5, 1$
σ, π	$19 + \|m\|, 1$	$120, 104$	$14, 4$	$69, 1$	$1620, 3$

5 Concluding Remarks

We compare our two constructions with some recent results in Table 1, taking both their security and efficiency into consideration. To compare with $\mathcal{SS}_1$ the signature scheme Σ is instantiated with Waters signature [30], the extractable public-key encryption scheme $\mathcal{E}$ is instantiated with Cramer-Shoup encryption [21], and the double-trapdoor chameleon hash is instantiated with the scheme by Chen *et al.* [20]. 'E' and 'P' denote the group exponentiation and pairing operation respectively. For $\mathcal{SS}_2$, Σ is instantiated with full Boneh-Boyen signature [5]. For simplicity, we do not differentiate between elements from different groups in this comparison. A more detailed comparison can be found in the full version. We remark that $\mathcal{SS}_2$ is instantiated with a composite order group. It shows that instantiating our generic construction leads to the first efficient unlinkable sanitizable signature schemes in the standard model.

Acknowledgments. We thank the anonymous reviewers for their comments on our manuscript.

Sherman Chow is supported by the Early Career Award and the Early Career Scheme (CUHK 439713), and General Research Funds (CUHK 14201914) of the Research Grants Council, University Grant Committee of Hong Kong.

Dominique Schröder is supported by the German Federal Ministry of Education and Research (BMBF) through funding for the project PROMISE and by the German research foundation (DFG) through funding for the collaborative research center 1223.

References

1. Abe, M., Chow, S.S.M., Haralambiev, K., Ohkubo, M.: Double-trapdoor anonymous tags for traceable signatures. Int. J. Inf. Sec. **12**(1), 19–31 (2013)
2. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable signatures. In: di Vimercati, S.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 159–177. Springer, Heidelberg (2005)
3. Backes, M., Meiser, S., Schröder, D.: Delegatable functional signatures. In: Cheng, C.-M., et al. (eds.) PKC 2016. LNCS, vol. 9614, pp. 357–386. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49384-7_14
4. Boldyreva, A., Palacio, A., Warinschi, B.: Secure proxy signature schemes for delegation of signing rights. J. Cryptology **25**(1), 57–115 (2012)
5. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
6. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J.: Foundations of fully dynamic group signatures. In: Manulis, M., Sadeghi, A.-R., Schneider, S. (eds.) ACNS 2016. LNCS, vol. 9696, pp. 117–136. Springer, Heidelberg (2016). doi:10.1007/978-3-319-39555-5_7
7. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on DDH. In: Pernul, G., et al. (eds.) ESORICS. LNCS, vol. 9326, pp. 243–265. Springer, Heidelberg (2015). doi:10.1007/978-3-319-24174-6_13
8. Bose, P., Das, D., Rangan, C.P.: Constant size ring signature without random oracle. In: Foo, E., Stebila, D. (eds.) ACISP 2015. LNCS, vol. 9144, pp. 230–247. Springer, Heidelberg (2015)
9. Brzuska, C., et al.: Redactable signatures for tree-structured data: definitions and constructions. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 87–104. Springer, Heidelberg (2010)
10. 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 (2009)
11. 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 (2010)
12. Brzuska, C., Pöhls, H.C., Samelin, K.: Non-interactive public accountability for sanitizable signatures. In: De Capitani di Vimercati, S., Mitchell, C. (eds.) EuroPKI 2012. LNCS, vol. 7868, pp. 178–193. Springer, Heidelberg (2013)
13. Brzuska, C., Pöhls, H.C., Samelin, K.: Efficient and perfectly unlinkable sanitizable signatures without group signatures. In: Katsikas, S., Agudo, I. (eds.) EuroMPI 2013. LNCS, vol. 8341, pp. 12–30. Springer, Heidelberg (2014)
14. Camenisch, J., Haralambiev, K., Kohlweiss, M., Lapon, J., Naessens, V.: Structure preserving CCA secure encryption and applications. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 89–106. Springer, Heidelberg (2011)
15. Canard, S., Jambert, A.: On extended sanitizable signature schemes. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 179–194. Springer, Heidelberg (2010)
16. Canard, S., Jambert, A., Lescuyer, R.: Sanitizable signatures with several signers and sanitizers. In: Mitrokotsa, A., Vaudenay, S. (eds.) AFRICACRYPT 2012. LNCS, vol. 7374, pp. 35–52. Springer, Heidelberg (2012)

17. Canard, S., Laguillaumie, F., Milhau, M.: *Trapdoor* sanitizable signatures and their application to content protection. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 258–276. Springer, Heidelberg (2008)
18. Catalano, D.: Homomorphic signatures and message authentication codes. In: Abdalla, M., De Prisco, R. (eds.) SCN 2014. LNCS, vol. 8642, pp. 514–519. Springer, Heidelberg (2014)
19. Catalano, D., Di Raimondo, M., Fiore, D., Gennaro, R.: Off-line/on-line signatures: theoretical aspects and experimental results. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 101–120. Springer, Heidelberg (2008)
20. Chen, X., Zhang, F., Tian, H., Wei, B., Susilo, W., Mu, Y., Lee, H., Kim, K.: Efficient generic on-line/off-line (threshold) signatures without key exposure. Inf. Sci. **178**(21), 4192–4203 (2008)
21. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998)
22. Derler, D., Slamanig, D.: Rethinking privacy for extended sanitizable signatures and a black-box construction of strongly private schemes. In: Au, M.-H., et al. (eds.) ProvSec 2015. LNCS, vol. 9451, pp. 455–474. Springer, Heidelberg (2015). doi:10.1007/978-3-319-26059-4_25
23. 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., et al. (eds.) PKC 2016. LNCS, vol. 9614, pp. 301–330. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49384-7_12
24. Furukawa, J., Yonezawa, S.: Group signatures with separate and distributed authorities. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 77–90. Springer, Heidelberg (2005)
25. Groth, J.: Fully anonymous group signatures without random oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
26. 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 (2008)
27. Johnson, R., Molnar, D., Song, D., Wagner, D.: Homomorphic signature schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
28. Klonowski, M., Lauks, A.: Extended sanitizable signatures. In: Rhee, M.S., Lee, B. (eds.) ICISC 2006. LNCS, vol. 4296, pp. 343–355. Springer, Heidelberg (2006)
29. Krawczyk, H., Rabin, T.: Chameleon signatures. In: NDSS 2000. The Internet Society, February 2000
30. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
31. Xu, S., Yung, M.: Accountable ring signatures: a smart card approach. In: Quisquater, J.-J., Paradinas, P., Deswarte, Y., El Kalam, A.A. (eds.) Smart Card Research and Advanced Applications VI. IFIP, vol. 153, pp. 271–286. Springer, Heidelberg (2004)