A General Framework for Redactable Signatures and New Constructions

David Derler[1,‡], Henrich C. Pöhls[2,‡,§], Kai Samelin[3,‖], Daniel Slamanig[1,‡]

[1] IAIK, Graz University of Technology, Austria
{david.derler|daniel.slamanig}@tugraz.at

[2] Institute of IT-Security and Security Law & Chair of IT-Security, University of Passau, Germany
hp@sec.uni-passau.de

[3] IBM Research – Zurich, Switzerland & TU Darmstadt, Germany
ksa@zurich.ibm.com

Abstract. A redactable signature scheme (RSS) allows *removing* parts of a signed message by any party without invalidating the respective signature. State-of-the-art constructions thereby focus on messages represented by one specific data-structure, e.g., lists, sets or trees, and adjust the security model accordingly. To overcome the necessity for this myriad of models, we present a general framework covering arbitrary data-structures and even more sophisticated possibilities. For example, we cover fixed elements which must not be redactable and dependencies between elements. Moreover, we introduce the notion of designated redactors, i.e., the signer can give some extra information to selected entities which become redactors. In practice, this often allows to obtain more efficient schemes. We then present two RSSs; one for sets and one for lists, both constructed from any EUF-CMA secure signature scheme and indistinguishable cryptographic accumulators in a black-box way and show how the concept of designated redactors can be used to increase the efficiency of these schemes. Finally, we present a black-box construction of a designated redactor RSS by combining an RSS for sets with non-interactive zero-knowledge proof systems. All the three constructions presented in this paper provide transparency, which is an important property, but quite hard to achieve, as we also conceal the length of the original message and the positions of the redactions.

1 Introduction

A redactable signature scheme (RSS) allows any party to *remove* parts of a signed message such that the corresponding signature σ can be updated without the signers' secret key sk. The so derived signature σ' then still verifies under the signer's public key pk. Hence, RSSs partially solve the "digital message sanitization problem" [MSI+03]. This separates RSSs from standard digital signatures, which prohibit *any* alteration of signed messages. Such a primitive comes in handy

This is the full version of a paper to appear at ICISC 2015.

‡ Supported by EU H2020 project PRISMACLOUD, grant agreement n°644962.
§ Supported by EU FP7 project RERUM, grant agreement n°609094.
‖ Supported by EU FP7 project FUTUREID, grant agreement n°318424.

in use-cases where only parts of the signed data are required, but initial origin authentication must still hold and re-signing is not possible or too expensive. One real-world application scenario is privacy-preserving handling of patient data [BBM09, BB12, SR10, WHT+10]. For instance, identifying information in a patient's record can be redacted for processing during accounting.

State-of-the-Art. RSSs have been introduced in [JMSD02, SB01]. Their ideas have been extended to address special data-structures such as trees [BBD+10, SPB+12a] and graphs [KB13]. While the initial idea was that redactions are public, the notion of accountable RSSs appeared recently [PS15]. Here, the redactor becomes a designated party which can be held accountable for redactions. Further, RSSs with dependencies between elements have been introduced and discussed in [BBM09]. Unfortunately, their work neither introduces a formal security model nor provides a security analysis for their construction. Consecutive redaction control allows intermediate redactors to prohibit further redactions by subsequent ones [MHI06, MIM+05, SPB+12b].

Much more work on RSSs exists. However, they do not use a common security model and most of the presented schemes do not provide the important security property denoted as transparency [BBD+10]. As an example, [HHH+08, KB13, WHT+10] are not transparent in our model. In such non-transparent constructions, a third party can potentially deduce statements about the original message from a redacted message-signature pair. In particular, their schemes allow to see where a redaction took place. Hence, they contradict the very intention of RSSs being a tool to increase or keep data privacy [BBD+10].

Ahn et al. [ABC+12] introduced the notion of statistically unlinkable RSSs as a stronger privacy notion. Their scheme only allows for quoting instead of arbitrary redactions, i.e., redactions are limited to the beginning and the end of an ordered list. Moreover, [ABC+12] only achieves the weaker and less common notion of selective unforgeability. Lately, even stronger privacy notions have been proposed in [ALP12, ALP13] in the context of the framework of $\mathcal{P}$-homomorphic signatures. There also exists a huge amount of related yet different signature primitives, where we refer the reader to [DDH+15] for a comprehensive overview of the state-of-the-art.

Motivation. RSSs have many applications. In particular, minimizing signed data before passing it to other parties makes RSSs an easy to comprehend privacy enhancing tool. However, the need for different security models and different data structures prohibits an easy integration into applications that require such privacy features, as RSSs do not offer a flexible, widely applicable framework. While the model of RSSs for sets (e.g. [MHI06]) can protect unstructured data such as votes, it is, e.g., unclear if it can be used for multi-sets. For ordered lists (such as a text) this already becomes more difficult: should one only allow quoting (i.e., redactions at the beginning and/or the end of a text) or general redactions? For trees (such as data-bases, XML or JSON), we have even more possibilities: only allow leaf-redactions [BBD+10], or leaves and inner nodes [KB13], or even allow to alter the structure [PSdMP12]. Furthermore, over the years more sophisticated features such as dependencies, fixed elements and redactable structure appeared. They complicate the specialized models even more.

We want to abandon the necessity to invent specialized security models tailored to specific use-cases and data-structures. Namely, we aim for a framework

that generalizes away details and covers existing approaches. Thereby, we want to keep the model compact, understandable and rigid. We aim at RSSs to become generally applicable to the whole spectrum of existing use-cases. In addition, we explicitly want to support the trend to allow the signer to limit the power of redactors [KL06, CJ10, DS15]. To prove the applicability of our framework, we present three new constructions which hide the length of the original message, the positions of redactions, and the fact that a redaction has even happened.

Contribution. Our contribution is manifold. (1) Existing work focuses on messages representations in only a specific data-structure, whereas our model is generally applicable (even for data-structures not yet considered for RSSs in the literature). Our general framework also captures more sophisticated redaction possibilities such as dependencies between redactable parts, fixed parts and consecutive redaction control. (2) We introduce the notion of designated redactors. While this concept might seem similar to the concept of accountable RSSs [PS15], we are not interested in accountability, but only want to allow to hand an extra piece of information to the redactor(s). This often allows to increase the efficiency of the respective scheme. (3) We present two RSSs, one for sets and one for lists, constructed in a black-box way from digital signatures and indistinguishable cryptographic accumulators. We show that existing constructions of RSSs are instantiations of our generic constructions but tailored to specific instantiations of accumulators (often this allows to optimize some of the parameters of the schemes). (4) We present a black-box construction of RSSs with designated redactors for lists from RSSs for sets and non-interactive zero-knowledge proof systems. We stress that all three proposed constructions provide transparency, which is an important property, but quite hard to achieve.

Notation. We use $\lambda \in \mathbb{N}$ to denote a security parameter and assume that all algorithms implicitly take 1^λ as an input. We write $y \leftarrow A(x)$ to denote the assignment of the output of algorithm A on input x to y. If we want to emphasize that A receives explicit random coins r, we write $y \leftarrow A(x;r)$. If S is a finite set, then $s \xleftarrow{R} S$ means that s is assigned a value chosen uniformly at random from S. We call an algorithm efficient, if it runs in probabilistic polynomial time (PPT) in the size of its input. Unless stated otherwise, all algorithms are PPT and return a special error symbol $\perp \notin \{0,1\}^*$ during an exception. A function $\epsilon : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is negligible, if it vanishes faster than every inverse polynomial. That is, for every $k \in \mathbb{N}$ there exists a $n_0 \in \mathbb{N}$ such that $\epsilon(n) \leq n^{-k}$ for all $n > n_0$. If the message M is a list, i.e., $M = (m_1, m_2, \ldots, m_{|M|})$, where $m_i \in \{0,1\}^*$, we call m_i a block. $|M| \in \mathbb{N}$ then denotes the number of blocks in the message M.

2 Generic Formalization of Redactable Signatures

This section presents our generalized definitions for RSSs.

2.1 The Generalized Framework

We use the formalization by *Brzuska* et al. [BBD+10] as a starting point. In contrast to their model, however, ours is not specifically tailored to trees, but is generally applicable to all kinds of data. The resulting model is rigid, i.e., it is more restrictive

than the ones introduced in the original works [JMSD02, SB01], while it is not as restrictive as [ABC+12, ALP12, ALP13, BFLS10, BPS13, CDHK15]. We think that the security model introduced in [BBD+10] is sufficient for most use cases, while the ones introduced in [ABC+12, ALP12, ALP13, BFLS10, BPS13, CDHK15] seem to be overly strong for real world applications. Namely, we require an RSS to be correct, unforgeable, private, and transparent. We explicitly do not require unlinkability and its derivatives (constituting even stronger privacy notations), as almost all messages (documents) occurring in real world applications contain data usable to link them, e.g., unique identifiers.[1] Moreover, we do not formalize accountability, as this notion can easily be achieved by applying the generic transformation presented in [PS15] to constructions being secure in our model.[2]

In the following, we assume that a message M is some arbitrarily structured piece of data and for the general framework we use the following notation. ADM is an abstract data structure which describes the admissible redactions and may contain descriptions of dependencies, fixed elements or relations between elements. MOD is used to actually describe how a message M is redacted. Next, we define how ADM, MOD and the message M are tangled, for which we introduce the following notation: $\mathsf{MOD} \overset{\mathsf{ADM}}{\preceq} M$ means that MOD is a valid redaction description with respect to ADM and M. $\mathsf{ADM} \preceq M$ denotes that ADM matches M, i.e., ADM is valid with respect to M. By $M' \xleftarrow{\mathsf{MOD}} M$, we denote the derivation of M' from M with respect to MOD. Clearly, how MOD, ADM, $\overset{\mathsf{ADM}}{\preceq}$, $\xleftarrow{\mathsf{MOD}}$ and $\preceq$ are implemented depends on the data structure in question and on the features of the concrete RSS. Let us give a simple example for sets without using dependencies or other advanced features: then, MOD and ADM, as well as M, are sets. A redaction $M' \xleftarrow{\mathsf{MOD}} M$ simply would be $M' \leftarrow M \setminus \mathsf{MOD}$. This further means that $\mathsf{MOD} \overset{\mathsf{ADM}}{\preceq} M$ holds if $\mathsf{MOD} \subseteq \mathsf{ADM} \subseteq M$, while $\mathsf{ADM} \preceq M$ holds if $\mathsf{ADM} \subseteq M$. We want to stress that the definitions of these operators also define how a redaction is actually performed, e.g., if a redacted block leaves a visible special symbol $\perp$ or not.

Now, we formally define an RSS within our general framework.

Definition 1. *An* RSS *is a tuple of four efficient algorithms* (KeyGen, Sign, Verify, Redact)*, which are defined as follows:*

$\mathsf{KeyGen}(1^\lambda)$: *On input of a security parameter λ, this probabilistic algorithm outputs a keypair* $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: *On input of a secret key* sk*, a message M and* ADM*, this (probabilistic) algorithm outputs a message-signature pair (M, σ) together with some auxiliary redaction information* red.[3]

$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: *On input of a public key* pk*, a signature σ and a message M, this deterministic algorithm outputs a bit $b \in \{0, 1\}$.*

$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: *This (probabilistic) algorithm takes a public key* pk*, a valid signature σ for a message M, modification instructions* MOD *and auxiliary redaction information* red *as input. It returns a redacted message-signature pair (M', σ') and an updated auxiliary redaction information* red′.[4]

[1] However, we stress that our model can be extended in a straightforward way.

[2] Our model could also be extended to cover accountability in a straightforward way.

[3] We assume that ADM can always be correctly and unambiguously derived from any valid message-signature pair. Also note that ADM may change after a redaction.

[4] Note that this algorithm may either explicitly or implicitly alter ADM in an unambiguous way.

We also require that Sign returns $\perp$, if $\mathsf{ADM} \not\preceq M$, while Redact also returns $\perp$, if $\mathsf{MOD} \overset{\mathsf{ADM}}{\not\preceq} M$. We will omit this explicit check in our constructions. Note that red can also be $\emptyset$ if no auxiliary redaction information is required.

2.2 Security Properties

The security properties for RSSs have already been formally treated for tree data-structures in [BBD+10]. We adapt them to our general framework.

Correctness. Correctness requires that all honestly computed/redacted signatures verify correctly. More formally this means that $\forall\ \lambda \in \mathbb{N},\ \forall\ n \in \mathbb{N},\ \forall\ M,\ \forall\ \mathsf{ADM} \preceq M, \forall\ (\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda),\ \forall\ ((M_0,\sigma_0),\mathsf{red}_0) \leftarrow \mathsf{Sign}(\mathsf{sk},M,\mathsf{ADM}),\ [\forall\ \mathsf{MOD}_i \overset{\mathsf{ADM}}{\preceq} M_i,\ \forall\ ((M_{i+1},\sigma_{i+1}),\mathsf{red}_{i+1}) \leftarrow \mathsf{Redact}(\mathsf{pk},\sigma_i,M_i,\mathsf{MOD}_i,\mathsf{red}_i)]_{0\le i<n}$ it holds that for $0 \le i \le n : \mathsf{Verify}(\mathsf{pk},\sigma_i,M_i) = 1$, where $[S_i]_{0\le i<n}$ is shorthand for $S_0,\ldots,S_{n-1}$.

Unforgeability. Unforgeability requires that without a signing key sk, it should be infeasible to compute a valid signature σ on a message M, which is not a valid redaction of any message obtained by adaptive signature queries.

Definition 2 (Unforgeability). *An* RSS *is unforgeable, if for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda), (M^*,\sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{Sign}}(\mathsf{sk},\cdot,\cdot)}(\mathsf{pk}) : \\ \mathsf{Verify}(\mathsf{pk},M^*,\sigma^*) = 1 \ \wedge\ M^* \notin Q_{\mathsf{Sign}}\end{array}\right] \le \epsilon(\lambda)$$

holds. $\mathcal{O}^{\mathsf{Sign}}$ denotes a signing oracle and we define $Q_{\mathsf{Sign}} \leftarrow \bigcup_{i=1}^{q}\{M' \mid M' \overset{\mathsf{MOD}_j}{\leftarrow\!\!\!\!-} M_i\ \forall\ \mathsf{MOD}_j \overset{\mathsf{ADM}_i}{\preceq} M_i\}$. Here, $q \in \mathbb{N}$ is the number of signing queries and M_i and ADM_i denote the respective input to $\mathcal{O}^{\mathsf{Sign}}$.

Note that an adversary can perform redactions on its own (also transitively).

Privacy. For anyone except the involved signers and redactors, it should be infeasible to derive information on redacted message parts when given a redacted message-signature pair.

Definition 3 (Privacy). *An* RSS *is private, if for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda), b \xleftarrow{R} \{0,1\}, \mathcal{O} \leftarrow \{\mathcal{O}^{\mathsf{Sign}}(\mathsf{sk},\cdot,\cdot), \\ \mathcal{O}^{\mathsf{LoRRedact}}((\mathsf{sk},\mathsf{pk}),\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,b)\}, b^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}) : b = b^*\end{array}\right] \le \frac{1}{2} + \epsilon(\lambda)$$

holds. $\mathcal{O}^{\mathsf{Sign}}$ is defined as before and $\mathcal{O}^{\mathsf{LoRRedact}}$ is defined as follows:

$\mathcal{O}^{\mathsf{LoRRedact}}((\mathsf{sk},\mathsf{pk}),M_0,\mathsf{MOD}_0,M_1,\mathsf{MOD}_1,\mathsf{ADM}_0,\mathsf{ADM}_1,b)$**:**

1: *Compute* $((M_c,\sigma_c),\mathsf{red}_c) \leftarrow \mathsf{Sign}(\mathsf{sk},M_c,\mathsf{ADM}_c)$ *for* $c \in \{0,1\}$.
2: *Let* $((M'_c,\sigma'_c),\mathsf{red}'_c) \leftarrow \mathsf{Redact}(\mathsf{pk},\sigma_c,M_c,\mathsf{MOD}_c,\mathsf{red}_c)$ *for* $c \in \{0,1\}$.
3: *If* $M'_0 \neq M'_1 \ \vee\ \mathsf{ADM}'_0 \neq \mathsf{ADM}'_1$, *return* $\perp$.
4: *Return* (M'_b,σ'_b).

Here, ADM'_0 and ADM'_1 are extracted from (M'_0,σ'_0) and (M'_1,σ'_1) and the oracle returns $\perp$ if any of the algorithms returns $\perp$.

In our privacy definition, we allow the adversary to provide distinct values for ADM_0 and ADM_1 to the signing oracle. While this guarantees the required flexibility to support arbitrary data structures, it yields a rather strong definition of privacy.

Transparency. It should be infeasible to decide whether a signature directly comes from the signer (i.e., is a fresh signature) or has been generated using the Redact algorithm, for anyone except the signer and the possibly involved redactor(s). More formally, this means:

Definition 4 (Transparency). *An* RSS *is transparent, if for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\epsilon(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda), b \xleftarrow{R} \{0,1\}, \mathcal{O} \leftarrow \{\mathcal{O}^{\mathsf{Sign}}(\mathsf{sk},\cdot,\cdot),\\ \mathcal{O}^{\mathsf{Sign/Redact}}((\mathsf{sk},\mathsf{pk}),\cdot,\cdot,\cdot,b)\}, b^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}) : b = b^*\end{array}\right] \leq \frac{1}{2} + \epsilon(\lambda)$$

holds. Here $\mathcal{O}^{\mathsf{Sign}}$ *is as in Definition 2 and* $\mathcal{O}^{\mathsf{Sign/Redact}}$ *is defined as follows:*

$\mathcal{O}^{\mathsf{Sign/Redact}}((\mathsf{sk},\mathsf{pk}), M, \mathsf{MOD}, \mathsf{ADM}, b)$**:**

1: *Compute* $((M,\sigma),\mathsf{red}) \leftarrow \mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$.
2: *Compute* $((M',\sigma_0),\mathsf{red}') \leftarrow \mathsf{Redact}(\mathsf{pk},\sigma,M,\mathsf{MOD},\mathsf{red})$.
3: *Compute* $((M',\sigma_1),\mathsf{red}'') \leftarrow \mathsf{Sign}(\mathsf{sk},M',\mathsf{ADM}')$.
4: *Return* (M',σ_b).

Note, ADM' *is extracted from* (M',σ_0) *and the oracle returns* $\perp$ *if any of the algorithms returns* $\perp$.

We call an RSS secure, if it is correct, unforgeable, private, and transparent.

We want to emphasize that additionally returning auxiliary redaction information red in Sign and Redact does not contradict transparency or privacy, as the "final" verifier never sees any red (which is why the privacy and transparency games do not return red for the challenge message-signature pair). Intuitively, only if an intermediate redactor exists, red is given away by the signer to selected designated entities that become redactors.[5]

Relations between Security Properties. The relations between the different security properties do not change compared to the work done in [BBD+10]. Namely, transparency implies privacy, while privacy does not imply transparency. Furthermore, unforgeability is independent of privacy and transparency. We prove these statements in Appendix B.

Notes on Our Model. In a nutshell, our generalized framework leaves the concrete data-structure—and, thus, also the definition of ADM, MOD, and red—open to the instantiation. For clarity, let us match our framework to already existing definitions. In particular, consider the model of [BBD+10]. It does not explicitly define ADM, but implicitly assumes that only leaves of a given tree are redactable, i.e., MOD may only contain changes which are possible with recursive leaf-redaction. *Pöhls* et al. [PSdMP12] explicitly define ADM as the edges between different nodes in their model for RSS for trees, while allowing arbitrary redactions, i.e., MOD may contain any set of nodes in the tree (including the tree's root), as well as edges.

[5] This also distinguishes designated redactors from accountable redactable signatures [PS15]. Namely, the additional information red can be given to any redactor, while the redactor is a fixed entity in accountable RSSs. Hence, in our notion, the redactors can even form a chain, and can be pinpointed in an ad-hoc manner.

Finally, we note that our model also covers consecutive redaction control [MHI06, MIM+05, SPB+12b] via ADM. Recall that ADM is contained in all signatures and Redact may also change ADM.

3 Building Blocks

In this section we provide the definitions of the required building blocks.

Digital Signature Schemes. We start by defining digital signatures.

Definition 5 (Digital Signatures). *A digital signature scheme* DSS *is a triple* (DKeyGen, DSign, DVerify) *of PPT algorithms:*

$\mathsf{DKeyGen}(1^\lambda)$: *This probabilistic algorithm takes a security parameter λ as input and outputs a secret (signing) key* sk *and a public (verification) key* pk *with associated message space $\mathcal{M}$.*[6]

$\mathsf{DSign}(\mathsf{sk}, m)$: *This (probabilistic) algorithm takes a message $m \in \mathcal{M}$ and a secret key* sk *as input, and outputs a signature σ.*

$\mathsf{DVerify}(\mathsf{pk}, m, \sigma)$: *This deterministic algorithm takes a signature σ, a message $m \in \mathcal{M}$ and a public key* pk *as input, and outputs a bit $b \in \{0, 1\}$.*

A DSS is secure, if it is correct and EUF-CMA secure. The formal security definitions are provided in Appendix A.1.

Cryptographic Accumulators. Cryptographic accumulators [BdM93] represent a finite set $\mathcal{X}$ as a single succinct value $\mathsf{acc}_\mathcal{X}$ and for each $x \in \mathcal{X}$ one can compute a witness wit_x, certifying membership of x in $\mathcal{X}$. We use the formal model from [DHS15] which assumes a trusted setup, i.e., a TTP generates the accumulator keypair $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$ and discards $\mathsf{sk}_{\mathsf{acc}}$. We, however, note that in some constructions $\mathsf{sk}_{\mathsf{acc}}$ improves efficiency, which is a useful feature if the party maintaining the accumulator is trusted (as it is the case in our schemes).[7]

In the formal model below, we omit some additional features of accumulators as they are not required here (cf. [DHS15]).

Definition 6 (Accumulator). *An accumulator* Acc *is a tuple of algorithms* (AGen, AEval,AWitCreate, AVerify) *which are defined as follows:*

$\mathsf{AGen}(1^\lambda, t)$: *This probabilistic algorithm takes a security parameter λ and a parameter t as input. If $t \neq \infty$, then t is an upper bound for the number of accumulated elements. It returns a key pair $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$, where $\mathsf{sk}_{\mathsf{acc}} = \emptyset$ if no trapdoor exists.*

$\mathsf{AEval}((\mathsf{sk}^{\sim}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X})$: *This (probabilistic) algorithm takes a key pair $(\mathsf{sk}^{\sim}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$ and a set $\mathcal{X}$ to be accumulated as input and returns an accumulator $\mathsf{acc}_\mathcal{X}$ together with some auxiliary information* aux.

$\mathsf{AWitCreate}((\mathsf{sk}^{\sim}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathsf{acc}_\mathcal{X}, \mathsf{aux}, x)$: *This (probabilistic) algorithm takes a key pair $(\mathsf{sk}^{\sim}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}})$, an accumulator $\mathsf{acc}_\mathcal{X}$, auxiliary information* aux *and a value x as input. It returns $\perp$, if $x \notin \mathcal{X}$, and a witness wit_x for x otherwise.*

[6] We usually omit to mention the message space $\mathcal{M}$ and assume that it is implicit in the public key.

[7] Such a trapdoor $\mathsf{sk}_{\mathsf{acc}}$, when used, does not influence the output distributions of the algorithms, but improves efficiency of some algorithms.

$\mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_x, x)$: *This deterministic algorithm takes a public key* $\mathsf{pk}_{\mathsf{acc}}$*, an accumulator* $\mathsf{acc}_{\mathcal{X}}$*, a witness* wit_x *and a value* x *as input and outputs a bit* $b \in \{0,1\}$*.*

An accumulator Acc is secure if it is correct, collision free, and indistinguishable. We recall the formal security definitions of these properties in Appendix A.2 and refer to [DHS15] for an overview of concrete instantiations. Henceforth, we use $\mathsf{Dom}(\mathsf{acc})$ to denote the accumulation domain.

Non-Interactive Commitments. We also require non-interactive commitment schemes, which we define below.

Definition 7 (Non-Interactive Commitment). *A non-interactive commitment scheme* Com *is a tuple of PPT algorithms* $(\mathsf{Gen}, \mathsf{Commit}, \mathsf{Open})$*, which are defined as follows:*

$\mathsf{Gen}(1^\lambda)$: *This probabilistic algorithm takes as input a security parameter* λ *and outputs the public parameters* pp *(subsequently, we omit* pp *for the ease of notation and assume that it is implicit input to all algorithms).*

$\mathsf{Commit}(m)$: *This (probabilistic) algorithm takes as input a message* m *and outputs a commitment* C *together with a corresponding opening information* O *including the randomness* r *used by* Commit*.*

$\mathsf{Open}(C, O)$: *This deterministic algorithm takes as input a commitment* C *with corresponding opening information* O *and outputs message* $m' \in m \cup \bot$*.*

A non-interactive commitment scheme Com is secure, if it is correct, (computationally) binding and (computationally) hiding. We provide a formal definition of the security properties in Appendix A.3. We call a commitment scheme homomorphic if for any m, m' we have $\mathsf{Commit}(m \oplus m') = \mathsf{Commit}(m) \otimes \mathsf{Commit}(m')$ for some binary operations $\oplus$ and $\otimes$. We emphasize that any perfectly correct IND-CPA secure public key encryption schemes yields perfectly binding commitments, e.g., ElGamal [Gam84], which is also homomorphic.

Non-Interactive Proof Systems. Now, we introduce non-interactive proofs for an **NP**-language with witness relation $R : L_R = \{x \mid \exists\, w : R(x, w) = 1\}$.

Definition 8 (Non-Interactive Proof System). *A non-interactive proof system* Π *is a tuple of algorithms* $(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$*, which are defined as follows:*

$\mathsf{Gen}_{\mathsf{crs}}(1^\lambda)$: *This probabilistic algorithm takes a security parameter* λ *as input, and outputs a common reference string* crs*.*

$\mathsf{Proof}(\mathsf{crs}, x, w)$: *This probabilistic algorithm takes a common reference string* crs*, a statement* x*, and a witness* w *as input, and outputs a proof* π*.*

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$: *This deterministic algorithm takes a common reference string* crs*, a statement* x*, and a proof* π *as input, and outputs* 1 *if* π *is valid and* 0 *otherwise.*

In our context, a non-interactive proof system Π is secure, if it is complete, sound, and adaptively zero-knowledge. We provide formal security definitions in Appendix A.4. Concrete instantiations of non-interactive proof systems, tailored to our requirements, are given in Section 5.

4 Redactable Signatures for Sets

For our RSS for sets (cf. Scheme 1), we compute an accumulator representing the set to be signed and then sign the accumulator using any digital signature scheme. For verification, one simply provides witnesses for each element in the set and it is verified whether the digital signature on the accumulator as well as the witnesses are valid. Redaction amounts to simply throwing away witnesses corresponding to redacted elements. To maintain transparency, while still allowing the signer to determine which blocks (i.e., elements) of the message (i.e., the set) are redactable, we model ADM as a set containing all blocks which must not be redacted. We also parametrize the scheme by an operator ord$(\cdot)$, which allows to uniquely encode ADM as a sequence. MOD is modeled as a set containing all blocks of the message to be redacted. We note that one can straightforwardly extend Scheme 1 to support

$\mathsf{KeyGen}(1^\lambda)$: This algorithm fixes a standard digital signature scheme DSS and an indistinguishable accumulator scheme $\mathsf{Acc} = \{\mathsf{AGen}, \mathsf{AEval}, \mathsf{AWitCreate}, \mathsf{AVerify}\}$, runs $(\mathsf{sk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{DSS}}) \leftarrow \mathsf{DKeyGen}(1^\lambda)$, $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda, \infty)$ and returns $(\mathsf{sk}, \mathsf{pk}) \leftarrow ((\mathsf{sk}_{\mathsf{DSS}}, \mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), (\mathsf{pk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{acc}}))$.

$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: This algorithm computes $(\mathsf{acc}_M, \mathsf{aux}) \leftarrow \mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), M)$, and for all $m_i \in M$: $\mathsf{wit}_{m_i} \leftarrow \mathsf{AWitCreate}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathsf{acc}_M, \mathsf{aux}, m_i)$. Finally, it computes $\sigma_{\mathsf{DSS}} \leftarrow \mathsf{DSign}(\mathsf{sk}_{\mathsf{DSS}}, \mathsf{acc}_M || \,\mathrm{ord}(\mathsf{ADM}))$ and returns (M, σ) and red, where $\sigma \leftarrow (\sigma_{\mathsf{DSS}}, \mathsf{acc}_M, \{\mathsf{wit}_{m_i}\}_{m_i \in M}, \mathsf{ADM})$ and $\mathsf{red} \leftarrow \emptyset$.

$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: This algorithms checks whether $\mathsf{DVerify}(\mathsf{pk}_{\mathsf{DSS}}, \mathsf{acc}_M || \,\mathrm{ord}(\mathsf{ADM}), \sigma_{\mathsf{DSS}}) = 1$, and for all $m_i \in M$: $\mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_M, \mathsf{wit}_{m_i}, m_i) = 1$. Furthermore, it checks whether $\mathsf{ADM} \cap M = \mathsf{ADM}$. It returns 1 if all checks hold and 0 otherwise.

$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: This algorithm parses σ as $(\sigma_{\mathsf{DSS}}, \mathsf{acc}_M, \mathsf{WIT}, \mathsf{ADM})$, computes $M' \leftarrow M \setminus \mathsf{MOD}$, sets $\mathsf{WIT}' \leftarrow \mathsf{WIT} \setminus \{\mathsf{wit}_{m_i}\}_{m_i \in \mathsf{MOD}}$ and returns (M', σ') and red′, where $\sigma' \leftarrow (\sigma_{\mathsf{DSS}}, \mathsf{acc}_M, \mathsf{WIT}', \mathsf{ADM})$ and $\mathsf{red}' \leftarrow \emptyset$.

---

ord(ADM)**:** This operator takes a set ADM, applies some unique ordering (e.g., lexicographic) to the elements in ADM and returns the corresponding sequence.

Scheme 1: A RSS for Sets

multi-sets by concatenating a unique identifier to each set element. In Appendix C.1 we prove the following:

Theorem 1. *If* Acc *and* DSS *are secure, then Scheme 1 is secure.*

4.1 Observations and Optimizations

Depending on the properties of the used accumulator scheme, one can reduce the signature size from $\mathcal{O}(n)$ to $\mathcal{O}(1)$. The required properties are as follows:

(1) The accumulator scheme needs to support batch-membership verification. Formally, this means that there are two additional algorithms AWitCreateB and AVerifyB, which are defined as follows:
$\mathsf{AWitCreateB}((\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}), \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, \mathcal{Y})$ is an deterministic algorithm that takes a key pair $(\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}})$, an accumulator $\mathsf{acc}_{\mathcal{X}}$, auxiliary information aux and a set $\mathcal{Y}$. It returns $\perp$, if $\mathcal{Y} \not\subseteq \mathcal{X}$, and a witness $\mathsf{wit}_{\mathcal{Y}}$ for $\mathcal{Y}$ otherwise.

$\mathsf{AVerifyB}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}}, \mathsf{wit}_{\mathcal{Y}}, \mathcal{Y})$ is a deterministic algorithm that takes a public key $\mathsf{pk}_{\mathsf{acc}}$, an accumulator $\mathsf{acc}_{\mathcal{X}}$, a witness $\mathsf{wit}_{\mathcal{Y}}$ and a set $\mathcal{Y}$. It returns `true` if $\mathsf{wit}_{\mathcal{Y}}$ is a witness for $\mathcal{Y} \subseteq \mathcal{X}$ and `false` otherwise.

(2) The accumulator scheme fulfills the quasi-commutativity property, i.e., with ρ being a fixed randomness it holds that

$$\begin{aligned}&\forall\ (\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^{\lambda}), \forall\ \mathcal{X}, \forall\ x \in \mathcal{X}, \forall\ \mathcal{Y} \subset \mathcal{X},\\ &(\mathsf{acc}_{\mathcal{X}}, \mathsf{aux}) \leftarrow \mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X}; \rho) :\\ &\mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X} \setminus \mathcal{Y}; \rho) = \mathsf{AWitCreateB}((\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}), \mathsf{acc}_{\mathcal{X}}, \mathsf{aux}, \mathcal{Y}).\end{aligned}$$

(3) It is possible to publicly add values to an accumulator.

Refer to [DHS15, Table 1] for a list of accumulators providing the required properties. From (1), (2), and (3) it is straightforward to derive the following corollary:

Corollary 1. *For schemes fulfilling (1), (2), and (3), it holds that* $\forall \{x, y\} \subseteq \mathcal{X}$*, one can use* $\mathsf{wit}_{\{x\}\cup\{y\}}$ *and* $\mathsf{acc}_{\mathcal{X}}$ *to attest that* x *is a member of* $\mathsf{acc}_{\mathcal{X}\setminus\{y\}}$*. Furthermore, one can efficiently compute* $\mathsf{wit}_{\{x\}\cup\{y\}}$ *from* $\mathsf{wit}_{\{x\}}$ *and* y*.*

Then, only a single witness needs to be stored and verification is performed with respect to this witness. Redaction is performed by publicly updating the witness (can be interpreted as removing elements from the accumulator). Such a scheme generalizes the RSS for sets from [PSPdM12], which builds upon the RSA accumulator. For accumulator schemes where (3) does not hold, one can still obtain constant size signatures by setting $\mathsf{red} \leftarrow \mathsf{aux}$. Upon Redact, red is not updated.

Our construction may look similar to the one in [PS14]. However, in contrast to our construction, they require a rather specific definition of accumulators, which they call trapdoor accumulators. Trapdoor accumulators differ from conventional accumulators regarding their features and security properties. In particular, they need to support updates of the accumulated set without modifying the accumulator itself. Further, they require a non-standard property denoted as strong collision resistance, which can be seen as a combination of conventional collision resistance and indistinguishability. Clearly, such a specific accumulator model limits the general applicability.

5 Redactable Signatures for Linear Documents

We build our RSS for linear documents upon the RSS for sets presented in the previous section. From an abstract point of view, moving from sets to linear documents means to move from an unordered message to an ordered one. A naive approach to assign an ordering to the message blocks would be to concatenate each message block with its position in the message and insert these extended tuples into the accumulator. However, such an approach trivially contradicts transparency, since the positions of the messages would reveal if redactions have taken place. Thus, inspired by [CLX09], we choose some indistinguishable accumulator scheme and use accumulators to encode the positions. More precisely, with n being the number of message blocks, we draw a sequence of n uniformly random numbers $(r_j)_{j=1}^{n}$ from the accumulation domain. Then, for each message block m_i, $1 \leq i \leq n$, an accumulator acc_i containing $(r_j)_{j=1}^{i}$ is computed (i.e., acc_i contains i randomizers). Finally,

for each m_i, one appends $\mathsf{acc}_i||r_i$ and signs the so obtained set $\bigcup_{j=1}^{n}\{(m_i||\mathsf{acc}_i||r_i)\}$ using the RSS for sets. Upon verification, one simply verifies the signature on the set and checks for each i whether one can provide i valid witnesses for $(r_j)_{j=1}^{i}$ with respect to acc_i. Redaction again amounts to throwing away witnesses corresponding to redacted message blocks.

Here, $M = (m_i)_{i=1}^{n}$ is a sequence of message blocks m_i, ADM is the corresponding sequence of fixed message blocks, and the operator $\mathrm{ord}(\cdot)$ for the underlying RSS for sets simply returns ADM without modification. All possible valid redactions, forming the transitive closure of a message M, with respect to Redact, are denoted as $\mathrm{span}_{\vdash}(M)$, following [CLX09] and [SPB+12b]. Note that for ADM it must hold that $\mathsf{ADM} \in \mathrm{span}_{\vdash}(M)$. MOD is modeled as a sequence of message blocks to be redacted and we assume an encoding that allows to uniquely match message block with its corresponding message block in the original message.

$\mathsf{KeyGen}(1^\lambda)$: This algorithm fixes a redactable signature scheme $\mathbf{RS}(\mathrm{ord}) = \{\mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verify}, \mathbf{Redact}\}$ for sets (with $\mathrm{ord}(\cdot)$ as defined below) and an indistinguishable accumulator scheme $\mathsf{Acc} = \{\mathsf{AGen}, \mathsf{AEval}, \mathsf{AWitCreate}, \mathsf{AVerify}\}$, runs $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda, \infty)$, $(\mathbf{sk}, \mathbf{pk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$ and returns $(\mathsf{sk}, \mathsf{pk}) \leftarrow ((\mathbf{sk}, \mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), (\mathbf{pk}, \mathsf{pk}_{\mathsf{acc}}))$.

$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: This algorithm chooses $(r_i)_{i=1}^{|M|} \xleftarrow{R} \mathsf{Dom}(\mathsf{acc})^{|M|}$, sets $M' \leftarrow \emptyset$ and computes for all r_i:

$$(\mathsf{acc}_i, \mathsf{aux}) \leftarrow \mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \cup_{j=1}^{i}\{r_j\}), \mathsf{WIT}_i \leftarrow (\mathsf{wit}_{ij})_{j=1}^{i}, \text{ where } \mathsf{wit}_{i_j} \leftarrow \mathsf{AWitCreate}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathsf{acc}_i, \mathsf{aux}, r_j).$$

Then it computes $\hat{\sigma} \leftarrow \mathbf{Sign}(\mathbf{sk}, \bigcup_{i=1}^{|M|}\{(m_i||\mathsf{acc}_i||r_i)\}, \mathsf{ADM})$. Finally, it returns (M, σ) and red, where $\sigma \leftarrow (\hat{\sigma}, (\mathsf{acc}_i)_{i=1}^{|M|}, (\mathsf{WIT}_i)_{i=1}^{|M|}, (r_i)_{i=1}^{|M|})$ and $\mathsf{red} \leftarrow \emptyset$.

$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: This algorithm checks whether $\mathbf{Verify}(\mathbf{pk}, \bigcup_{i=1}^{|M|}\{(m_i||\mathsf{acc}_i||r_i)\}, \hat{\sigma}) = 1$. Furthermore, it verifies for all $1 \le i \le |M|$ whether $(\mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_i, \mathsf{wit}_{i_j}, r_j) = 1)_{j=1}^{i}$. Finally it checks whether $\mathsf{ADM} \in \mathrm{span}_{\vdash}(M)$. If all checks hold it returns 1 and 0 otherwise.

$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: This algorithm sets $\mathsf{MOD}' \leftarrow \emptyset$ and for all $m_i \in \mathsf{MOD}$: $\mathsf{MOD}' \leftarrow \mathsf{MOD}' \cup \{(m_i||\mathsf{acc}_i||r_i)\}$ runs $(\cdot, \hat{\sigma}') \leftarrow \mathbf{Redact}(\mathbf{pk}, \hat{\sigma}, \bigcup_{i=1}^{|M|}\{(m_i||\mathsf{acc}_i||r_i)\}, \mathsf{MOD}')$. Then for all $m_i \in \mathsf{MOD}$ it removes the corresponding entries from M, $(\mathsf{acc}_i)_{i=1}^{|M|}, (\mathsf{WIT}_i)_{i=1}^{|M|}$ and $(r_i)_{i=1}^{|M|}$ and obtains M', $(\mathsf{acc}_i)_{i=1}^{|M'|}, (\mathsf{WIT}_i)_{i=1}^{|M'|}$ and $(r_i)_{i=1}^{|M'|}$. Finally, it returns (M', σ') and red', where $\sigma' \leftarrow (\hat{\sigma}', (\mathsf{acc}_i)_{i=1}^{|M'|}, (\mathsf{WIT}_i)_{i=1}^{|M'|}, (r_i)_{i=1}^{|M'|})$ and $\mathsf{red}' \leftarrow \emptyset$.

$\mathrm{ord}(\mathsf{ADM})$**:** This operator returns ADM.

Scheme 2: A RSS for Linear Documents

Theorem 2. *If* Acc *and* **RS** *are secure, then Scheme 2 is secure.*

We prove Theorem 2 in Appendix C.2.

5.1 Observations and Optimizations

Depending on the used accumulator scheme, it is possible to reduce the signature size from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.Let us assume that (1), (2), and (3) from Section 4.1 hold, which means that also Corollary 1 holds. Then, due to (1), one only needs to store one witness $\mathsf{wit}_{\bigcup_{k=1}^{i}\{r_k\}}$ per message block, where i is the position of the block in the message. Furthermore, upon redaction of message block i with corresponding randomizer r_i, one can update the witnesses $\mathsf{wit}_{\bigcup_{k=1}^{j}\{r_j\}}$ for all $i > j \leq |M|$ by computing $\mathsf{wit}'_{r_j} \leftarrow \mathsf{wit}_{\bigcup_{k=1}^{j}\{r_j\}\cup\{r_i\}}$ and removing witness $\mathsf{wit}_{\bigcup_{k=1}^{i}\{r_i\}}$ and randomizer r_i from the signature, which boils down to removing r_i from all accumulators. The so-obtained construction then essentially generalizes the approach of [CLX09], which make (white-box) use of the RSA accumulator. If (3) does not hold, one can use a similar strategy as in Section 4.1.

5.2 RSS for Linear Documents with Designated Redactors

The signature size and computational complexity of RSSs can often be improved by explicitly considering the possibility to allow red to be non-empty. In Scheme 3 we follow this approach and present such a generic construction of RSSs for linear documents. Basically, the idea is to compute commitments to the positions of the messages blocks and concatenate them to the respective message blocks. Then, one signs the so obtained set of concatenated messages and commitments using an RSS for sets. Additionally, one includes a non-interactive zero-knowledge proof of an order relation on the committed positions for attesting the correct order of the message blocks. The information red then represents the randomness used to compute the single commitments. Redacting message blocks then simply amounts to removing the single blocks from the signature of the RSS for sets and recomputing a non-interactive proof for the ordering on the remaining commitments. Since redaction control via ADM can straightforwardly be achieved as in Scheme 2, we omit it here for simplicity, i.e., we assume $\mathsf{ADM} = \infty$. Also note that without ADM the operator $\mathrm{ord}(\cdot)$ is not required. MOD is defined as in Scheme 2. We emphasize that one can easily obtain constant size red by pseudorandomly generating the randomizers $(r_i)_{i=1}^{|M|}$ and storing the seed for the PRG in red instead of the actual randomizers.

Instantiating proof system Π for R_{ord} can be done straightforwardly by using zero-knowledge set membership proofs. Below, we briefly discuss the efficiency of the instantiations of Scheme 3, when based on three common techniques. We note that the below Σ-protocols can all easily be made non-interactive (having all the required properties) using the Fiat-Shamir transform.

Square Decomposition. An efficient building block for range proofs in hidden order groups is a proof that a secret integer x is positive [Bou00, Lip03], which is sufficient for our instantiation. Technically, therefore we need an homomorphic integer commitment scheme and R_{ord} for Π is as follows:

$$((C_1, C_2), (x, r)) \in R_{\mathsf{ord}} \iff C_2 - C_1 = \mathsf{Commit}(x; r) \ \wedge \ x \geq 0.$$

This approach yields $O(n)$ signature generation cost, signature size and verification cost and has a constant size public key. It, however, only works in a hidden order group setting.

$\mathsf{KeyGen}(1^\lambda)$: This algorithm fixes a redactable signature scheme for sets $\mathbf{RS} = \{\mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verify}, \mathbf{Redact}\}$, a commitment scheme $\mathsf{Com} = (\mathsf{Gen}, \mathsf{Commit}, \mathsf{Open})$ as well as a non-interactive zero-knowledge proof system $\Pi = (\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ for the following **NP**-relation R_{ord} with $O_i = (x_i, r_i)$:

$$((C_1, C_2), (O_1, O_2)) \in R_{\mathsf{ord}} \iff C_1 = \mathsf{Commit}(x_1; r_1) \ \wedge \ C_2 = \mathsf{Commit}(x_2; r_2) \ \wedge \ x_1 \leq x_2.$$

It runs $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$, $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathsf{crs} \leftarrow \mathsf{Gen}_{\mathsf{crs}}(1^\lambda)$, sets $(\mathsf{sk}, \mathsf{pk}) \leftarrow ((\mathbf{sk}, \mathsf{pp}, \mathsf{crs}), (\mathbf{pk}, \mathsf{pp}, \mathsf{crs}))$ and returns $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: If $\mathsf{ADM} \neq \infty$, this algorithm returns $\perp$. Otherwise, it sets $D \leftarrow \emptyset$ and computes for $1 \leq i \leq |M|$: $(C_i, O_i) \leftarrow \mathsf{Commit}(\mathsf{pp}, i)$, $D \leftarrow D \cup \{(C_i || m_i)\}$ and for $1 \leq i < |M|$: $\pi_i \leftarrow \mathsf{Proof}(\mathsf{crs}, (C_i, C_{i+1}), (O_i, O_{i+1}))$. Then, it computes $\hat{\sigma} \leftarrow \mathbf{Sign}(\mathbf{sk}, D, \infty)$ and returns (M, σ), where $\sigma \leftarrow (\hat{\sigma}, (C_i)_{i=1}^{|M|}, (\pi_i)_{i=1}^{|M|-1})$ as the signature and $\mathsf{red} \leftarrow (O_i)_{i=1}^{|M|}$ as private information for the redactor.[a]

$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: This algorithms sets $D \leftarrow \emptyset$ and for all $m_i \in M : D \leftarrow D \cup \{(C_i || m_i)\}$. It checks whether $\mathbf{Verify}(\mathbf{pk}, D, \hat{\sigma}) = 1$. Furthermore, for $1 \leq i < |M|$ it checks whether $\mathsf{Verify}(\mathsf{crs}, (C_i, C_{i+1}), \pi_i) = 1$. If any of the checks fails it returns 0 and 1 otherwise.

$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: This algorithm sets $\mathsf{MOD}' \leftarrow \emptyset$ and for all $m_i \in \mathsf{MOD} : \mathsf{MOD}' \leftarrow \mathsf{MOD}' \cup \{(C_i || m_i)\}$. Then it runs $(\cdot, \hat{\sigma}') \leftarrow \mathbf{Redact}(\mathbf{pk}, \hat{\sigma}, \bigcup_{i=1}^{|M|}\{(C_i || m_i)\}, \mathsf{MOD}')$. Then for all $m_i \in \mathsf{MOD}$ it removes the corresponding entries from M, $(C_i)_{i=1}^{|M|}$ and $(O_i)_{i=1}^{|M|}$ to obtain M', $(C_i)_{i=1}^{|M'|}$ and $(O_i)_{i=1}^{|M'|}$. In the end, it computes for $1 \leq i < |M'| : \pi_i \leftarrow \mathsf{Proof}(\mathsf{crs}, (C_i, C_{i+1}), (O_i, O_{i+1}))$, sets $\sigma' \leftarrow (\hat{\sigma}', (C_i)_{i=1}^{|M'|}, (\pi_i)_{i=1}^{|M'|-1})$, $\mathsf{red}' \leftarrow (O_i)_{i=1}^{|M'|}$ and returns (M', σ') and red'.

[a] We note that we set $\mathsf{red} \leftarrow (O_i)_{i=1}^{|M|} = (m_i, r_i)_{i=1}^{|M|}$ for notational convenience, while one would only require $\mathsf{red} \leftarrow (r_i)_{i=1}^{|M|}$.

Scheme 3: A Designated Redactor RSS for Linear Documents

For the subsequent two approaches we need to introduce an upper bound k on the number of message blocks which will be a parameter of Scheme 3.

Multi-Base Decomposition. This technique for range proofs works by decomposing the secret integer $x = \sum_{i=1}^{n} G_i \cdot b_i$ with $b_i \in [0, u-1]$ into a (multi)-base representation and then proving that every b_i belongs to the respective small set ([LAN02], cf. [CCJT13] for an overview). It also works in the prime order group setting. Here, the relation R_{ord} for Π is as follows:

$$((C_1, C_2), (x, r)) \in R_{\mathsf{ord}} \iff C_2 - C_1 = \mathsf{Commit}(x; r) \ \wedge \ 0 \leq x < k.$$

This approach yields $O(n \log k)$ signature generation costs, signature size and verification costs and a constant size public key.

Signature-Based Approach. This technique [CCS08] pursues the idea of signing every element in the interval[8] using a suitable signature scheme $(\mathsf{DKeyGen}, \mathsf{DSign}, \mathsf{DVerify})$. In our application let the interval be $[0, k[$ and let us denote the corresponding public signatures by $\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{k-1})$. Now, proving membership of x in $[0, k[$ amounts to the relation R_{ord} under crs being σ and the respective

[8] Actually, [CCS08] also propose a combination of this approach with a (multi)-base decomposition, which we do not consider here for brevity.

public key pk_σ (public parameters):

$$((C_1, C_2), (x, r)) \in R_{\mathsf{ord}} \iff C_2 - C_1 = \mathsf{Commit}(x; r) \ \wedge \ \exists\, i \in [0, k[\ : \mathsf{DVerify}(\mathsf{pk}_\sigma, x, \sigma_i) = 1.$$

This approach yields $O(n)$ signature generation cost, signature size and verification cost. The crs representing the public signatures and the verification key may be included into the public key of **RS**, yielding a public key of size $O(k)$.

Finally, we prove Theorem 3 in Appendix C.3.

Theorem 3. *If* Com*,* Π*, and* **RS** *are secure, then Scheme 3 is secure.*

References

[ABC+12] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, A. Shelat, and B. Waters. Computing on Authenticated Data. In *TCC*, pages 1–20, 2012.

[ALP12] N. Attrapadung, B. Libert, and T. Peters. Computing on authenticated data: New privacy definitions and constructions. In *ASIACRYPT*, pages 367–385, 2012.

[ALP13] N. Attrapadung, B. Libert, and T. Peters. Efficient completely context-hiding quotable and linearly homomorphic signatures. In *PKC*, pages 386–404, 2013.

[BB12] J. Brown and D. M. Blough. Verifiable and redactable medical documents. In *AMIA*, 2012.

[BBD+10] C. Brzuska, H. Busch, Ö. Dagdelen, M. Fischlin, M. Franz, S. Katzenbeisser, M. Manulis, C. Onete, A. Peter, B. Poettering, and D. Schröder. Redactable Signatures for Tree-Structured Data: Definitions and Constructions. In *ACNS*, pages 87–104, 2010.

[BBM09] D. Bauer, D. M. Blough, and A. Mohan. Redactable signatures on data with dependencies and their application to personal health records. In *WPES*, pages 91–100, 2009.

[BdM93] J. Benaloh and M. de Mare. One-way accumulators: a decentralized alternative to digital signatures. In *EUROCRYPT*, pages 274–285, 1993.

[BFF+09] C. Brzuska, M. Fischlin, T. Freudenreich, A. Lehmann, M. Page, J. Schelbert, D. Schröder, and F. Volk. Security of Sanitizable Signatures Revisited. In *PKC*, pages 317–336. Springer, 2009.

[BFLS10] C. Brzuska, M. Fischlin, A. Lehmann, and D. Schröder. Unlinkability of Sanitizable Signatures. In *PKC*, pages 444–461, 2010.

[BGI14] E. Boyle, S. Goldwasser, and I. Ivan. Functional Signatures and Pseudorandom Functions. In *PKC*, pages 501–519, 2014.

[Bou00] F. Boudot. Efficient Proofs that a Committed Number Lies in an Interval. In *EUROCRYPT*, pages 431–444, 2000.

[BPS13] C. Brzuska, H. C. Pöhls, and K. Samelin. Efficient and perfectly unlinkable sanitizable signatures without group signatures. In *EuroPKI*, pages 12–30, 2013.

[CCJT13] S. Canard, I. Coisel, A. Jambert, and J. Traoré. New results for the practical use of range proofs. In *EuroPKI*, pages 47–64, 2013.

[CCS08] J. Camenisch, R. Chaabouni, and A. Shelat. Efficient Protocols for Set Membership and Range Proofs. In *ASIACRYPT*, pages 234–252, 2008.

[CDHK15] J. Camenisch, M. Dubovitskaya, K. Haralambiev, and M. Kohlweiss. Composable & modular anonymous credentials: Definitions and practical constructions. *IACR Cryptology ePrint Archive*, 2015:580, 2015.

[CJ10] S. Canard and A. Jambert. On extended sanitizable signature schemes. In *CT-RSA*, pages 179–194, 2010.

[CLX09] E.-C. Chang, C. L. Lim, and J. Xu. Short redactable signatures using random trees. In *CT-RSA*, pages 133–147, 2009.

[DDH+15] D. Demirel, D. Derler, C. Hanser, H. C. Pöhls, D. Slamanig, and G. Traverso. PRISMACLOUD D4.4: Overview of Functional and Malleable Signature Schemes. Technical report, H2020 Prismacloud, www.prismacloud.eu, 2015.

[DHS15] D. Derler, C. Hanser, and D. Slamanig. Revisiting cryptographic accumulators, additional properties and relations to other primitives. In *CT-RSA*, pages 127–144, 2015.

[DS15] D. Derler and D. Slamanig. Rethinking Privacy for Extended Sanitizable Signatures and a Black-Box Construction of Strongly Private Schemes. In *ProvSec*, 2015. Full Version: IACR Cryptology ePrint Report 2015/843.

[Gam84] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[GMR88] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM JoC*, 17(2):281–308, 1988.

[HHH+08] S. Haber, Y. Hatano, Y. Honda, W. G. Horne, K. Miyazaki, T. Sander, S. Tezoku, and D. Yao. Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In *ASIACCS*, pages 353–362, 2008.

[JMSD02] R. Johnson, D. Molnar, D. Song, and D.Wagner. Homomorphic signature schemes. In *CT-RSA*, pages 244–262, 2002.

[KB13] A. Kundu and E. Bertino. Privacy-preserving authentication of trees and graphs. *Int. J. Inf. Sec.*, 12(6):467–494, 2013.

[KL06] M. Klonowski and A. Lauks. Extended Sanitizable Signatures. In *ICISC*, pages 343–355, 2006.

[LAN02] H. Lipmaa, N. Asokan, and V. Niemi. Secure Vickrey Auctions without Threshold Trust. In *Financial Cryptography*, pages 87–101, 2002.

[Lip03] H. Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In *ASIACRYPT*, pages 398–415, 2003.

[MHI06] K. Miyazaki, G. Hanaoka, and H. Imai. Digitally signed document sanitizing scheme based on bilinear maps. In *ASIACCS*, pages 343–354, 2006.

[MIM+05] K. Miyazaki, M. Iwamura, T. Matsumoto, R. Sasaki, H. Yoshiura, S. Tezuka, and H. Imai. Digitally signed document sanitizing scheme with disclosure condition control. *IEICE Transactions*, 88-A(1):239–246, 2005.

[MSI+03] K. Miyazaki, S. Susaki, M. Iwamura, T. Matsumoto, R. Sasaki, and H. Yoshiura. Digital documents sanitizing problem. *IEICE Technical Report, ISEC2003-20*, 2003.

[PS14] H. C. Pöhls and K. Samelin. On updatable redactable signatures. In *ACNS*, pages 457–475, 2014.

[PS15] H. C. Pöhls and K. Samelin. Accountable redactable signatures. In *ARES*, pages 60–69, 2015.

[PSdMP12] H. C. Pöhls, K. Samelin, H. de Meer, and J. Posegga. Flexible redactable signature schemes for trees - extended security model and construction. In *SECRYPT 2012*, pages 113–125, 2012.

[PSPdM12] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Length-hiding redactable signatures from one-way accumulators in $O(n)$. Technical report, 2012.

[SB01] R. Steinfeld and L. Bull. Content extraction signatures. In *ICISC*, pages 285–304, 2001.

[SPB+12a] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. On Structural Signatures for Tree Data Structures. In *ACNS*, volume 7341 of *LNCS*, pages 171–187. Springer, 2012.

[SPB+12b] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. Redactable signatures for independent removal of structure and content. In *ISPEC*, pages 17–33, 2012.

[SR10] D. Slamanig and S. Rass. Generalizations and extensions of redactable signatures with applications to electronic healthcare. In *CMS*, pages 201–213, 2010.

[WHT+10] Z.-Y. Wu, C.-W. Hsueh, C.-Y. Tsai, F. Lai, H.-C. Lee, and Y. Chung. Redactable Signatures for Signed CDA Documents. *Journal of Medical Systems*, pages 1–14, 2010.

A Security Models

A.1 Digital Signatures

A digital signature scheme DSS is required to be correct, i.e., for all security parameters $\lambda \in \mathbb{N}$, all $(\mathsf{sk}, \mathsf{pk})$ generated by $\mathsf{DKeyGen}$ and all $m \in \mathcal{M}$ one requires $\mathsf{DVerify}(\mathsf{pk}, m, \mathsf{DSign}(\mathsf{sk}, m)) = 1$. Furthermore, we require existential unforgeability under adaptively chosen-message attacks ($\mathsf{EUF\text{-}CMA}$ security) [GMR88].

Definition 9 ($\mathsf{EUF\text{-}CMA}$). *A* DSS *is* $\mathsf{EUF\text{-}CMA}$ *secure, if for all PPT adversaries* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\left[\begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{DKeyGen}(1^\lambda), (m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{DSign}(\mathsf{sk},\cdot)}}(\mathsf{pk}) : \\ \mathsf{DVerify}(\mathsf{pk}, m^*, \sigma^*) = 1 \ \wedge \ m^* \notin Q^{\mathsf{DSign}}\end{array}\right] \leq \epsilon(\lambda) \ ,$$

where $\mathcal{A}$ *has access to an oracle* $\mathcal{O}^{\mathsf{DSign}}$ *that allows to execute the* DSign *algorithm and the environment keeps track of all message queried to* $\mathcal{O}^{\mathsf{DSign}}$ *via* Q^{DSign}.

A.2 Indistinguishable Accumulators

While correctness is omitted because it is straightforward, we recall the definition for collision freeness and indistinguishability from [DHS15] below.

Definition 10 (Collision Freeness). *An accumulator is* collision-free*, if for all PPT adversaries* $\mathcal{A}$ *and all* t *there is a negligible function* $\epsilon(\cdot)$ *such that:*

$$\Pr\left[\begin{array}{c}(\mathsf{sk}_{\mathsf{acc}},\mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda, t), \ \mathcal{O} \leftarrow \{\mathcal{O}^{\mathsf{E}(\cdot,\cdot,\cdot)}, \mathcal{O}^{\mathsf{W}(\cdot,\cdot,\cdot,\cdot)}\}, \\ (\mathsf{wit}_x^*, x^*, \mathcal{X}^*, \rho^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}_{\mathsf{acc}}) : \\ \mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}^*, \mathsf{wit}_x^*, x^*) = 1 \ \wedge \ x^* \notin \mathcal{X}^*\end{array}\right] \leq \epsilon(\lambda),$$

where $\mathsf{acc}^* \leftarrow \mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X}^*; \rho^*)$*. Here,* $\mathcal{O}^{\mathsf{E}}$ *and* $\mathcal{O}^{\mathsf{W}}$ *represent the oracles for the algorithms* AEval *and* $\mathsf{AWitCreate}$*, respectively. In case of randomized accumulators, the adversary also outputs the used randomness* ρ^**. Likewise, the adversary can control the randomness* r *used by* $\mathcal{O}^{\mathsf{E}}$ *for randomized accumulators.*

Definition 11 (Indistinguishability). *An accumulator is* indistinguishable*, if for all PPT adversaries* $\mathcal{A}$ *and all* t *there is a negligible function* $\epsilon(\cdot)$ *such that:*

$$\Pr\left[\begin{array}{c}(\mathsf{sk}_{\mathsf{acc}},\mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda, t), \ b \xleftarrow{R} \{0,1\}, \ (\mathcal{X}_0, \mathcal{X}_1, \\ \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pk}_{\mathsf{acc}}), (\mathsf{acc}_{\mathcal{X}_b}, \mathsf{aux}) \leftarrow \mathsf{AEval}((\mathsf{sk}_{\mathsf{acc}}^{\sim}, \mathsf{pk}_{\mathsf{acc}}), \mathcal{X}_b), \\ \mathcal{O} \leftarrow \{\mathcal{O}^{\mathsf{E}(\cdot,\cdot,\cdot)}, \mathcal{O}^{\mathsf{W}(\cdot,\cdot,\mathsf{aux},\cdot)}\}, \ b^* \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{\mathcal{X}_b}, \mathsf{state}) : \\ b = b^*\end{array}\right] \leq \frac{1}{2} + \epsilon(\lambda),$$

where $\mathcal{X}_0$ and $\mathcal{X}_1$ are two distinct subsets of the accumulation domain. Here, $\mathcal{O}^{\mathsf{E}}$ is defined as before, whereas $\mathcal{O}^{\mathsf{W}}$ is restricted to queries for values $x \in \mathcal{X}_0 \cap \mathcal{X}_1$. Furthermore, the input parameter aux *for $\mathcal{O}^{\mathsf{W}}$ is kept up to date and is provided by the environment, since $\mathcal{A}$ could trivially distinguish using* aux*.*

A.3 Non-Interactive Commitments

While correctness is straightforward and therefore omitted, the remaining security properties of non-interactive commitments are defined as follows.

Definition 12 (Binding). *A non-interactive commitment scheme is* binding*, if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda), (C^*, O^*, O'^*) \leftarrow \mathcal{A}(\mathsf{pp}), m \leftarrow \mathsf{Open}(C^*, O^*), \\ m' \leftarrow \mathsf{Open}(C^*, O'^*) : m \neq m' \quad \wedge \quad m \neq \bot \quad \wedge \quad m' \neq \bot\end{array}\right] \leq \epsilon(\lambda).$$

If $\epsilon = 0$, a commitment scheme is called perfectly binding.

Definition 13 (Hiding). *A non-interactive commitment scheme is* hiding*, if for all PPT adversaries $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\left[\begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda), (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp}), b \xleftarrow{R} \{0,1\}, \\ (C, O) \leftarrow \mathsf{Commit}(m_b), b^* \leftarrow \mathcal{A}(\mathsf{pp}, C, \mathsf{state}) : \\ b = b^*\end{array}\right] \leq \frac{1}{2} + \epsilon(\lambda).$$

A.4 Non-Interactive Proof Systems

Subsequently, we present the security properties for non-interactive proof systems which are required in our context (adapted from [BGI14]). Therefore, let L_R be an **NP**-language with witness relation $R : L_R = \{x \mid \exists\, w : R(x, w) = 1\}$.

Definition 14 (Completeness). *A non-interactive proof system* $(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ *is* complete*, if for every adversary $\mathcal{A}$ it holds that*

$$\Pr\Big[\mathsf{crs} \leftarrow \mathsf{Gen}_{\mathsf{crs}}(1^\lambda), (x, w) \leftarrow \mathcal{A}(\mathsf{crs}), \pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x, w) : \\ \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \ \wedge \ (x, w) \in R\Big] = 1.$$

Definition 15 (Soundness). *A non-interactive proof system* $(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ *is* sound*, if for every PPT adversary $\mathcal{A}$ there is a negligible function $\epsilon(\cdot)$ such that*

$$\Pr\Big[\mathsf{crs} \leftarrow \mathsf{Gen}_{\mathsf{crs}}(1^\lambda), (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) : \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \ \wedge \ x \notin L_R\Big] \leq \epsilon(\lambda).$$

$(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ *is perfectly sound, if $\epsilon = 0$.*

Definition 16 (Adaptive Zero-Knowledge). *A non-interactive proof system* $(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ *is* adaptively zero-knowledge*, if there exists a simulator* $S = (S_1, S_2)$ *such that for every PPT adversary* $\mathcal{A}$ *there is a negligible function* $\epsilon(\cdot)$ *such that*

$$\Big|\Pr\Big[\mathsf{crs} \leftarrow \mathsf{Gen}_{\mathsf{crs}}(1^\lambda) : \mathcal{A}^{\mathcal{P}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) = 1\Big] - \Pr\Big[(\mathsf{crs}, \tau_s) \leftarrow S_1(1^\lambda) : \mathcal{A}^{\mathcal{S}(\mathsf{crs},\tau_s,\cdot,\cdot)}(\mathsf{crs}) = 1\Big]\Big| \leq \epsilon(\lambda),$$

where, τ_s *denotes a simulation trapdoor. Thereby,* $\mathcal{P}$ *and* $\mathcal{S}$ *return* $\perp$ *if* $(x, w) \notin R$ *or* $\pi \leftarrow \mathsf{Proof}(\mathsf{crs}, x, w)$ *and* $\pi \leftarrow S_2(\mathsf{crs}, \tau_s, x)$*, respectively, otherwise.* $(\mathsf{Gen}_{\mathsf{crs}}, \mathsf{Proof}, \mathsf{Verify})$ *is perfect adaptively zero-knowledge, if* $\epsilon = 0$*.*

B Relations between Security Properties

The following relations were already given by *Brzuska* et al. [BBD+10].

Proposition 1 (Transparency $\implies$ Privacy). *Every transparent* RSS *is also private.*

Proposition 2 (Privacy $\not\Longrightarrow$ Transparency). *Not every private* RSS *is also transparent.*

Proposition 3 (Unforgeability is Independent). *Not every unforgeable* RSS *is private.*

We now prove Propositions 1-3. Our proofs are essentially the same as given in [BBD+10] resp. in [BFF+09], but adjusted to our framework.

Proof (of Proposition 1). Assume an efficient adversary $\mathcal{A}^{\mathsf{priv}}$ that wins the privacy game with probability $1/2 + \epsilon(\lambda)$, where $\epsilon(\cdot)$ is non-negligible. We construct an efficient adversary $\mathcal{A}^{\mathsf{tran}}$ which wins the transparency game with probability $1/2 + \epsilon(\lambda)/2$. Subsequently, we describe an efficient reduction $\mathcal{R}^{\mathsf{tran}}$ that interacts with a transparency challenger $\mathcal{C}^{\mathsf{tran}}$, such that $(\mathcal{R}^{\mathsf{tran}}, \mathcal{A}^{\mathsf{priv}})$ form $\mathcal{A}^{\mathsf{tran}}$.

$\mathcal{R}^{\mathsf{tran}}$ receives the public key pk from $\mathcal{C}^{\mathsf{tran}}$ and is given oracle access to $\mathcal{O}^{\mathsf{Sign}}$ and $\mathcal{O}^{\mathsf{Sign/Redact}}$. $\mathcal{R}^{\mathsf{tran}}$ tosses a coin $b' \xleftarrow{R} \{0,1\}$ and initializes $\mathcal{A}^{\mathsf{priv}}$ with pk. It is easy to see that $\mathcal{R}^{\mathsf{tran}}$ can answer each signing query of $\mathcal{A}^{\mathsf{priv}}$ by using the signing oracle provided by $\mathcal{C}^{\mathsf{tran}}$. For each query $(M_0, \mathsf{MOD}_0, M_1, \mathsf{MOD}_1, \mathsf{ADM}_0, \mathsf{ADM}_1)$ of $\mathcal{A}^{\mathsf{priv}}$ to $\mathcal{O}^{\mathsf{LoRRedact}}$, $\mathcal{R}^{\mathsf{tran}}$ checks whether $M'_0 = M'_1$, where $M'_0 \xleftarrow{\mathsf{MOD}_0} M_0$ and $M'_1 \xleftarrow{\mathsf{MOD}_1} M_1$. If so, it forwards $(M_b, \mathsf{MOD}_b, \mathsf{ADM}_b)$ to $\mathcal{O}^{\mathsf{Sign/Redact}}$ of $\mathcal{C}^{\mathsf{tran}}$ and returns the result. Otherwise, it returns $\perp$. At some point, $\mathcal{A}^{\mathsf{priv}}$ outputs its guess b^*. $\mathcal{R}^{\mathsf{tran}}$ returns 0, if $b^* = b'$, and 1 otherwise.

If $b = 0$, then $\mathcal{O}^{\mathsf{Sign/Redact}}$ always redacts. Hence, the view of of $\mathcal{A}^{\mathsf{priv}}$ is the same as in the privacy game. If, however, $b = 1$, then each signature is fresh and the output of $\mathcal{A}^{\mathsf{priv}}$ does not help to win the transparency game. It follows that $\Pr[\mathcal{R}^{\mathsf{tran}} = b] = 1/2(1/2 + \epsilon(\lambda)) + 1/2 \cdot 1/2 = 1/2 + \epsilon(\lambda)/2$. □

Proof (of Proposition 2). Assume an unforgeable, private, and transparent $\mathsf{RSS} = (\mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verify}, \mathbf{Redact})$. We modify it in the following way to obtain $\mathsf{RSS}' = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact})$.

$\mathsf{KeyGen}(1^\lambda)$: Return $\mathbf{KeyGen}(1^\lambda)$.
$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: Run $((M', \sigma), \mathsf{red}) \leftarrow \mathbf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$ and return $((M', \sigma \| 0), \mathsf{red})$.
$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: Parse σ as $\sigma' \| b, b \in \{0,1\}$ and return $\mathbf{Verify}(\mathsf{pk}, \sigma', M)$.
$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: Parse σ as $\sigma' \| b, b \in \{0,1\}$, run $((M', \sigma''), \mathsf{red}') \leftarrow \mathbf{Redact}(\mathsf{pk}, \sigma', M, \mathsf{MOD}, \mathsf{red})$ and return $((M', \sigma'' \| 1), \mathsf{red}')$.

It is easy to see that privacy and unforgeability of RSS carry over to RSS', while transparency trivially does not hold anymore. □

Proof (of Proposition 3). Assume an unforgeable, private, and transparent $\mathsf{RSS} = (\mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verify}, \mathbf{Redact})$. We modify it in the following way to obtain $\mathsf{RSS}' = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact})$.

$\mathsf{KeyGen}(1^\lambda)$: Return $\mathbf{KeyGen}(1^\lambda)$.
$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: Return $\mathbf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$
$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: Return 1.
$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: Return $\mathbf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$.

The contrived scheme is still transparent and therefore private, while producing a forgery is trivial.

For the other direction, we modify RSS as follows and obtain $\mathsf{RSS}'' = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact})$.

$\mathsf{KeyGen}(1^\lambda)$: Return $\mathbf{KeyGen}(1^\lambda)$.
$\mathsf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$: Run $((M', \sigma), \mathsf{red}) \leftarrow \mathbf{Sign}(\mathsf{sk}, M, \mathsf{ADM})$ and return $((M', \sigma || M), \mathsf{red})$.
$\mathsf{Verify}(\mathsf{pk}, \sigma, M)$: Parse σ as $\sigma' || M$ and return $\mathbf{Verify}(\mathsf{pk}, \sigma', M)$.
$\mathsf{Redact}(\mathsf{pk}, \sigma, M, \mathsf{MOD}, \mathsf{red})$: Parse σ as $\sigma' || M'$, run $((M'', \sigma''), \mathsf{red}') \leftarrow \mathbf{Redact}(\mathsf{pk}, \sigma', M, \mathsf{MOD}, \mathsf{red})$ and return $((M'', \sigma'' || M'), \mathsf{red}')$.

Clearly, unforgeability is still given, while the scheme is obviously not private (and, thus, not transparent) due to the original M being available. □

C Security Proofs

C.1 Proof of Theorem 1

We show that Theorem 1 holds by proving Lemma 1-3 and deriving Corollary 2.

Lemma 1. *If* Acc *is correct and* DSS *is correct, the construction in Scheme 1 is correct.*

The lemma above follows from inspection.

Lemma 2. *If* Acc *is collision free and* DSS *is existentially unforgeable under chosen-message attacks, the construction in Scheme 1 is unforgeable.*

Proof. Assume an efficient adversary $\mathcal{A}^{\mathsf{uf}}$ against unforgeability. We show how $\mathcal{A}^{\mathsf{uf}}$ can be used to construct (1) an efficient adversary $\mathcal{A}^{\mathsf{cf}}$ against the collision freeness of the accumulator or (2) an efficient adversary $\mathcal{A}^{\mathsf{euf\text{-}cma}}$ against the $\mathsf{EUF\text{-}CMA}$ security of the signature scheme. To do so, we describe efficient reductions $\mathcal{R}^{\mathsf{cf}}$ and $\mathcal{R}^{\mathsf{euf\text{-}cma}}$, respectively.

1. Here, $\mathcal{R}^{\mathsf{cf}}$ obtains the accumulator public key $\mathsf{pk}_{\mathsf{acc}}$ from the challenger $\mathcal{C}^{\mathsf{cf}}$ of the collision freeness game of the used accumulator scheme and completes the setup by running $(\mathsf{sk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{DSS}}) \leftarrow \mathsf{DKeyGen}(1^\lambda)$ and handing $(\mathsf{pk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{acc}})$ to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that $\mathcal{R}^{\mathsf{cf}}$ can simulate all oracles for $\mathcal{A}^{\mathsf{uf}}$ by forwarding the respective calls to $\mathcal{O}^{\mathsf{E}}$ and $\mathcal{O}^{\mathsf{W}}$ provided by $\mathcal{C}^{\mathsf{cf}}$. Furthermore, $\mathcal{R}^{\mathsf{cf}}$ can choose the randomness used in the calls to $\mathcal{O}^{\mathsf{E}}$ and keeps a mapping of accumulators and corresponding randomizers. Eventually, $\mathcal{A}^{\mathsf{uf}}$ outputs a tuple (M^*, σ^*), where $\sigma^* = (\sigma^*_{\mathsf{DSS}}, \mathsf{acc}_{M^*}, \{\mathsf{wit}_{m_i}\}_{m_i \in M^*}, \mathsf{ADM}^*)$ such that $\nexists (M, \mathsf{ADM}^*) \in Q_{\mathsf{Sign}} \ \nexists \mathsf{MOD}^{\mathsf{ADM}^*}_{\preceq} M : M^* \xleftarrow{\mathsf{MOD}} M$. If $\mathsf{acc}_{M^*} || \operatorname{ord}(\mathsf{ADM}^*)$ was never signed using DSS, it aborts. Otherwise, we have at least one m_i with a corresponding witness wit_{m_i} such that $\mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_{M^*}, \mathsf{wit}_{m_i}, m_i) = 1$ but $m_i \notin M^*$. Consequently, $\mathcal{R}^{\mathsf{cf}}$ can look up the randomness ρ used to compute acc_{M^*} and output $(\mathsf{wit}_{m_i}, m_i, M^*, \rho)$ as a collision for the accumulator.
2. Here, $\mathcal{R}^{\mathsf{euf\text{-}cma}}$ obtains the DSS public key $\mathsf{pk}_{\mathsf{DSS}}$ from the challenger $\mathcal{C}^{\mathsf{euf\text{-}cma}}$ of the $\mathsf{EUF\text{-}CMA}$ game of the used signature scheme and completes the setup by running $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda)$ and handing $(\mathsf{pk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{acc}})$ to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that $\mathcal{R}^{\mathsf{cf}}$ can simulate all oracles for $\mathcal{A}^{\mathsf{uf}}$ by forwarding the respective calls to Sign to the DSign oracle provided by $\mathcal{C}^{\mathsf{euf\text{-}cma}}$. Eventually, $\mathcal{A}^{\mathsf{uf}}$ outputs a tuple (M^*, σ^*), where $\sigma^* = (\sigma^*_{\mathsf{DSS}}, \mathsf{acc}^*_M, \{\mathsf{wit}_{m_i}\}_{m_i \in M^*}, \mathsf{ADM}^*)$ such that $\nexists (M, \mathsf{ADM}^*) \in Q_{\mathsf{Sign}} \ \nexists \mathsf{MOD}^{\mathsf{ADM}^*}_{\preceq} M : M^* \xleftarrow{\mathsf{MOD}} M$. If $\mathsf{acc}_{M^*} || \operatorname{ord}(\mathsf{ADM}^*)$ was signed using the signing oracle provided by $\mathcal{C}^{\mathsf{euf\text{-}cma}}$ it aborts. Otherwise, we can output $(\sigma^*_{\mathsf{DSS}}, \mathsf{acc}_{M^*} || \operatorname{ord}(\mathsf{ADM}^*))$ as a forgery. □

Lemma 3. *If* Acc *is indistinguishable, the construction in Scheme 1 is transparent.*

Proof. We will bound the probability to win the transparency game by using a sequence of games. Thereby, we denote the event that the adversary wins Game i by S_i.

Game 0: The original transparency game.

Game 1: As the original game, but all calls to AEval in $\mathcal{O}^{\mathsf{Sign/Redact}}$ are performed with respect to the originally submitted message M.

Transition Game 0 → Game 1: By the indistinguishability property of the accumulator, the adversary will only detect the game change with negligible probability $k \cdot \epsilon_{\mathsf{ind}}(\lambda)$, where k is the number of accumulators.[9]

In Game 1, the value of the accumulator is independent of the bit b. The remaining signature components are identically distributed. This means that $\Pr[S_1] = \frac{1}{2}$, and, in further consequence, $\Pr[S_0] \leq \frac{1}{2} + k \cdot \epsilon_{\mathsf{ind}}(\lambda)$. □

The implication of privacy by transparency allows to derive the following corollary.

Corollary 2. *The construction in Scheme 1 is private.*

C.2 Proof of Theorem 2

We show that Theorem 2 holds by proving Lemma 4-6 and deriving Corollary 3.

[9] For compactness, we collapse the exchange of the accumulators to one single game change, which can straightforwardly be unrolled to k game changes.

Lemma 4. *If* Acc *is correct and* **RS** *is correct, the construction in Scheme 2 is correct as well.*

The lemma above follows from inspection.

Lemma 5. *If* Acc *is collision free and* **RS** *is unforgeable, the construction in Scheme 2 is unforgeable.*

Proof. Assume an efficient adversary $\mathcal{A}^{\mathsf{uf}}$ against unforgeability. We show how $\mathcal{A}^{\mathsf{uf}}$ can be used to construct (1) an efficient adversary $\mathcal{A}^{\mathsf{cf}}$ against the collision freeness of the accumulator or (2) an efficient adversary $\mathcal{A}^{\mathsf{uf}}$ against the unforgeability of the underlying RSS for sets **RS**. To do so, we describe efficient reductions $\mathcal{R}^{\mathsf{cf}}$ and $\mathcal{R}^{\mathsf{uf}}$, respectively.

1. Here, $\mathcal{R}^{\mathsf{cf}}$ obtains the accumulator public key $\mathsf{pk}_{\mathsf{acc}}$ from the challenger $\mathcal{C}^{\mathsf{cf}}$ of the collision freeness game of the used accumulator scheme and completes the setup by running $(\mathbf{sk}, \mathbf{pk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$ and handing $(\mathbf{pk}, \mathsf{pk}_{\mathsf{acc}})$ to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that $\mathcal{R}^{\mathsf{cf}}$ can simulate all oracles for $\mathcal{A}^{\mathsf{uf}}$ by forwarding the respective calls to $\mathcal{O}^{\mathsf{E}}$ and $\mathcal{O}^{\mathsf{W}}$ provided by $\mathcal{C}^{\mathsf{cf}}$. Furthermore, $\mathcal{R}^{\mathsf{cf}}$ can choose the randomness used in the calls to $\mathcal{O}^{\mathsf{E}}$ and keeps a mapping of accumulators and corresponding randomizers. Eventually, $\mathcal{A}^{\mathsf{uf}}$ outputs a tuple (M^*, σ^*), where $\sigma^* = (\hat{\sigma}^*, (\mathsf{acc}_i)_{i=1}^{|M^*|}, (\mathsf{WIT}_i)_{i=1}^{|M^*|}, (r_i)_{i=1}^{|M^*|})$ and $\hat{\sigma}^*$ contains ADM^* such that $\nexists (M, \mathsf{ADM}^*) \in Q_{\mathsf{Sign}}\ \nexists \mathsf{MOD}^{\preceq^{\mathsf{ADM}^*}} M : M^* \xleftarrow{\mathsf{MOD}} M$. If there was no signing query for a superset of $\bigcup_{i=1}^{|M^*|}\{(m_i||\mathsf{acc}_i||r_i)$ and ADM^*, it aborts. Otherwise, we have at least one accumulator acc_i, corresponding set $R_i = \{r_j\}_{j=1}^{i}$, witness wit_{r_k} and randomizer r_k such that $r_k \notin R_i$ but $\mathsf{AVerify}(\mathsf{pk}_{\mathsf{acc}}, \mathsf{acc}_i, \mathsf{wit}_{r_k}, r_k) = 1$. Then, $\mathcal{R}^{\mathsf{cf}}$ can look up the randomizer ρ corresponding to acc_i and output $(\mathsf{wit}_{r_k}, r_k, R_i, \rho)$ as a collision for the accumulator.
2. Here, $\mathcal{R}^{\mathsf{uf}}$ obtains the RSS public key $\mathbf{pk}$ from the challenger $\mathcal{C}^{\mathsf{uf}}$ of the unforgeability game of the used redactable signature scheme for sets and completes the setup by running $(\mathsf{sk}_{\mathsf{acc}}, \mathsf{pk}_{\mathsf{acc}}) \leftarrow \mathsf{AGen}(1^\lambda)$ and handing $(\mathsf{pk}_{\mathsf{DSS}}, \mathsf{pk}_{\mathsf{acc}})$ to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that $\mathcal{R}^{\mathsf{cf}}$ can simulate all oracles for $\mathcal{A}^{\mathsf{uf}}$ by forwarding the respective calls to **Sign** to the oracles provided by $\mathcal{C}^{\mathsf{uf}}$. Eventually, $\mathcal{A}^{\mathsf{uf}}$ outputs a tuple (M^*, σ^*), where $\sigma^* = (\hat{\sigma}^*, (\mathsf{acc}_i)_{i=1}^{|M^*|}, (\mathsf{WIT}_i)_{i=1}^{|M^*|}, (r_i)_{i=1}^{|M^*|})$ and $\hat{\sigma}^*$ contains ADM^* such that $\nexists (M, \mathsf{ADM}^*) \in Q_{\mathsf{Sign}}\ \nexists \mathsf{MOD}^{\preceq^{\mathsf{ADM}^*}} M : M^* \xleftarrow{\mathsf{MOD}} M$. If a superset of $\bigcup_{i=1}^{|M^*|}\{(m_i||\mathsf{acc}_i||r_i)\}$ and ADM^* was signed using the oracle provided by $\mathcal{C}^{\mathsf{uf}}$ it aborts. Otherwise, it outputs the tuple $(\hat{\sigma}^*, \bigcup_{i=1}^{|M^*|}\{(m_i||\mathsf{acc}_i||r_i))$ as a forgery for the underlying RSS for sets. □

Lemma 6. *If* Acc *is indistinguishable and* **RS** *is transparent, the construction in Scheme 2 is transparent.*

Proof. We will bound the probability to win the transparency game by using a sequence of games. Thereby, we denote the event that the adversary wins Game i by S_i.

Game 0: The original transparency game.

Game 1: As the original game, but all accumulators acc_i in $\mathcal{O}^{\mathsf{Sign/Redact}}$ are computed with respect to the initial set of randomizers $\{r_i\}_{i=1}^{|M|}$.

Transition Game 0 → Game 1: By the indistinguishability property of the accumulator, the adversary will only detect the game change with negligible probability

$k \cdot \epsilon_{\mathsf{ind}}(\lambda)$, where k is the number of accumulators.[10]
In Game 1, the accumulators acc_i are independent of the bit b. In this game the adversary can only win the game by breaking the transparency of the underlying RSS, i.e., $\Pr[S_1] \leq \frac{1}{2} + \epsilon_{\mathsf{RSS}}(\lambda)$. All in all, we have that $|\Pr[S_0] - \Pr[S_1]| \leq k \cdot \epsilon_{\mathsf{ind}}(\lambda)$, meaning that the probability to win the transparency game is $\Pr[S_0] \leq \frac{1}{2} + \epsilon_{\mathsf{RSS}}(\lambda) + k \cdot \epsilon_{\mathsf{ind}}(\lambda)$, which is negligible. □

The implication of privacy by transparency allows to derive the following corollary.

Corollary 3. *The construction in Scheme 2 is private.*

C.3 Proof of Theorem 3

We show that Theorem 3 holds by proving Lemma 7-9 and deriving Corollary 4.

Lemma 7. *If* **RS** *is correct,* Com *is correct, and* Π *is complete, the construction in Scheme 3 is correct.*

The lemma above follows from inspection.

Lemma 8. *If* **RS** *is unforgeable,* Com *is perfectly binding, and* Π *is sound, the construction in Scheme 3 is unforgeable.*

Proof. To prove unforgeability, we show how an efficient adversary against unforgeability $\mathcal{A}^{\mathsf{uf}}$ can be used to construct (1) an efficient adversary $\mathcal{A}^{\mathsf{uf}}$ against the unforgeability of the underlying RSS for sets **RS** or (2) an efficient adversary $\mathcal{A}^{\mathsf{so}}$ against the soundness of the underlying proof system. Subsequently, we describe efficient reductions $\mathcal{R}^{\mathsf{uf}}$ and $\mathcal{R}^{\mathsf{so}}$, respectively.

1. $\mathcal{R}^{\mathsf{uf}}$ obtains the RSS public key $\mathbf{pk}$ from the challenger $\mathcal{C}^{\mathsf{uf}}$ of the unforgeability game of the used redactable signature scheme for sets and completes the setup by running $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathsf{crs} \leftarrow \mathsf{Gen}_{\mathsf{crs}}(1^\lambda)$, setting $(\mathsf{sk}, \mathsf{pk}) \leftarrow ((\mathbf{sk}, \mathsf{pp}, \mathsf{crs}), (\mathbf{pk}, \mathsf{pp}, \mathsf{crs}))$ and handing pk to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that $\mathcal{R}^{\mathsf{cf}}$ can simulate all oracles for $\mathcal{A}^{\mathsf{uf}}$ by forwarding the respective calls to **Sign** to the oracles provided by $\mathcal{C}^{\mathsf{uf}}$. Eventually, $\mathcal{A}^{\mathsf{uf}}$ outputs a valid tuple (M^*, σ^*), where $\sigma^* = (\hat{\sigma}^*, (C_i)_{i=1}^{|M^*|}, (\pi_i)_{i=1}^{|M^*|-1})$ such that $\nexists (M, \infty) \in Q_{\mathsf{Sign}} : M^* \in \mathrm{span}_{\vdash}(M)$. If a superset of $\bigcup_{i=1}^{|M^*|} \{(C_i || m_i)\}$ was signed using the oracle provided by $\mathcal{C}^{\mathsf{uf}}$ it aborts. Otherwise, it outputs $(\hat{\sigma}^*, \bigcup_{i=1}^{|M|} \{(C_i || m_i)\})$ as a forgery for the RSS for sets.
2. $\mathcal{R}^{\mathsf{so}}$ obtains crs from the challenger $\mathcal{C}^{\mathsf{so}}$ of the soundness game of the underlying non-interactive proof system and completes the setup by running $(\mathbf{sk}, \mathbf{pk}) \leftarrow \mathbf{KeyGen}(1^\lambda)$, $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda)$, setting $(\mathsf{sk}, \mathsf{pk}) \leftarrow ((\mathbf{sk}, \mathsf{pp}, \mathsf{crs}), (\mathbf{pk}, \mathsf{pp}, \mathsf{crs}))$ and handing pk to $\mathcal{A}^{\mathsf{uf}}$. It is easy to see that the reduction can simulate all oracles as in the real game. Eventually, $\mathcal{A}^{\mathsf{bd}}$ outputs a valid tuple (M^*, σ^*), where $\sigma^* = (\hat{\sigma}^*, (C_i^*)_{i=1}^{|M^*|}, (\pi_i^*)_{i=1}^{|M^*|-1})$ such that $\nexists (M, \infty) \in Q_{\mathsf{Sign}} : M^* \in \mathrm{span}_{\vdash}(M)$. If no superset of $\bigcup_{i=1}^{|M^*|} \{(C_i || m_i)\}$ was ever signed using the oracle provided by $\mathcal{C}^{\mathsf{uf}}$ it aborts. Otherwise, there is at least one i for $1 \leq i < |M^*|$ such that

[10] As in the proof of Lemma 3, we collapse the exchange of the accumulators to one single game change for compactness.

$\mathsf{Verify}(\mathsf{crs}, (C_i^*, C_{i+1}^*), \pi_i) = 1$ but $(C_i^*, C_{i+1}^*) \notin L_{R_{\mathsf{ord}}}$, which means that $\mathcal{R}^{\mathsf{so}}$ can output $((C_i^*, C_{i+1}^*), \pi_i)$ to win the soundness game of the non-interactive proof system. □

Lemma 9. *If* **RS** *is transparent,* Com *is hiding, and Π is adaptively zero-knowledge, the construction in Scheme 3 is transparent.*

Proof. We will bound the probability to win the transparency game by using a sequence of games. Thereby, we denote the event that the adversary wins Game i by S_i.

Game 0: The original transparency game.

Game 1: As the original game, but the environment sets up the crs for the non-interactive proof system using the simulator S_1, i.e., $(\mathsf{crs}, \tau_s) \leftarrow S_1(1^\lambda)$ and followingly simulates all proofs using $S_2(\mathsf{crs}, \tau_s, \cdot, \cdot)$.

Transition Game 0 → Game 1: A distinguisher between Game 0 and Game 1 is a distinguisher for the adaptive zero-knowledge property of the underlying non-interactive proof system, i.e., the distinguishing probability $|\Pr[S_0] - \Pr[S_1]|$ is bounded by $\epsilon_{\mathsf{zk}}(\lambda)$.

Game 2: As Game 1, but all commitments inside $\mathcal{O}^{\mathsf{Sign/Redact}}$ are replaced by commitments to 0.

Transition Game 1 → Game 2: A distinguisher between Game 1 and Game 2 is a distinguisher for the hiding game of the underlying commitment scheme, i.e., the distinguishing probability $|\Pr[S_1] - \Pr[S_2]|$ is bounded by $|M'| \cdot \epsilon_{\mathsf{hd}}(\lambda)$.[11]

In Game 2, all values except $\hat{\sigma}$ are independent of the bit b, meaning that the adversary has the same advantage as in the privacy game of the underlying RSS for sets, i.e., $\Pr[S_1] = 1/2 + \epsilon_{\mathsf{RSS}}(\lambda)$. Taking all together, we have $\Pr[S_0] \leq 1/2 + \epsilon_{\mathsf{RSS}}(\lambda) + \epsilon_{\mathsf{zk}}(\lambda) + |M'| \cdot \epsilon_{\mathsf{hd}}(\lambda)$, which is negligible. □

The implication of privacy from transparency allows to derive the following corollary.

Corollary 4. *The construction in Scheme 3 is private.*

[11] For compactness, we combine the exchange of the commitments in one game change and note that it is straightforward to unroll the exchange of the commitments in $|M'|$ game changes.