Policy-Based Redactable Set Signatures

Zachary A. Kissel

Department of Computer and Data Sciences, Merrimack College
kisselz@merrimack.edu

Abstract

A redactable set signature scheme is a signature scheme that allows a redactor, without possessing the signing key, to convert a signature on set S to a signature on set S' if $S' \subset S$. This paper introduces a new form of redactable set signature scheme called a *policy-based redactable set signature scheme*. These redactable set signatures allow for a signer to provide a redaction policy at signing time that limits the possible redactions that can be made by a redactor. In particular, a signature on set S can only be redacted to a signature on S' if $S' \subset S$ and $P(S') = 1$.

1 Introduction

A *redactable signature scheme* (RSS)[JMSW02, SBZ01] is a signature scheme where the a signer may sign a message m (resulting in signature σ) in such a way that a third party may redact the original message m to m' updating signature σ to σ' without the signing key. Two types of message have been considered in the literature, vectors and sets. Applications of this primitive include redacting of sensitive of data from documents (vectors) [JMSW02, SBZ01] and anonymous credential systems (sets) [San20, CL01, CL02, Cha85]. One of the problems with redactable signature schemes, as pointed out in [BPS17], is known schemes allow for arbitrary redaction by a third party. In this work we make progress towards solving an open question of Bilzhausse et. al. We define, and provide constructions of, a policy-based redactable set signature scheme that allows the original signer to limit what can be redacted by a third party. We refer to such signature schemes as *policy-based redactable set signatures* (RSSS). In a policy-based RSSS, given only the verification key, the third party may redact set S to S' if and only if $S' \subset S$ and S' satisfies the policy associated with the *signature*, specified at *signing time*. Moreover, a third party may further restrict the policy during redaction to limit future redactions. The last property is not new, the notion of *controlled disclosure* of [MIM+05] is already present in some RSS schemes. We emphasize that our policy-based constructions additionally put redaction control in the hands of the signer, unlike controlled disclosure. Readers familiar with P-Homomorphic signatures [ABC+15] may see similarities. However, our formulation differs in that our policy is specified at signing time and has a role in redaction and verification.

1.1 Contributions

This work has two main contributions.

1. We give a formal definition of policy-based redactable set signatures along with associated security notions.
2. We give a several constructions of a policy-based redactable set signature that satisfy various guarantees.

1.2 Related Work

Early works [JMSW02, SBZ01, DPSS15, CLX09, ...] focused on two types of data: vector and sets. Additionally, some initial work has been undertaken on tree structured data [BBD+10]. A useful survey of the redactable signatures can be found in [BPS17]. Redactable signatures have three main forms of security a relaxed form of EUF-CMA security called unforgeability of RSS, privacy which ensures nothing is learned about redacted portions, and a security notion is unlinkability. The unlinkability notion says that is computationally infeasible for an adversary to link a redacted signature to the signature it was generated from. Early constructions did not consider unlinkability, however, the property became important once it was shown that unlinkable redactable signature schemes can be used to construct anonymous credential systems [CL01].

A trivial (and inefficient) vector construction was given by [JMSW02]. For an n dimensional vector $\vec{m} = (m_0, m_1, \ldots, m_{n-1})$ a redactable signature scheme can be constructed by signing each element of the vector. In particular, For each element i compute $\sigma_i = \mathsf{Sign}_{sk}(r \parallel i \parallel m_i)$, where $r \xleftarrow{\$} \{0,1\}^n$ is the same for all i. To redact, the ith element the redactor deletes m_i and signature σ_i. The trivial construction satisfies unforgeablilty and privacy[1]. By dropping the index of the message block from the signature, one can achieve a trivial construction that supports sets. This set construction is unforgeable and private.

Johnson et. al provide a more advanced vector construction [JMSW02] based on Merkle trees coupled with the GGM PRF[GGM86] tree construction. Roughly speaking, each element 0 through $n-1$ form the leaves of a GGM tree such that block m_i is paired with $k_i = f_s(i)$. Next, the hash of the $k_i \parallel m_i$ is computed for every block i and the results placed at the leaves of a Merkle tree. The signature is $\sigma = (s, v, \mathsf{Sign}_{sk}(v))$ where v is the root of the Merkle tree. The construction only offers unforgeability. Depending on the application this may offer sufficient security. Because of the dependence on Merkle trees, a set construction does not easily follow, unless one wants to restrict the notion to ordered sets.

Derler et. al [DPSS15] gave a set construction and a vector construction. The set construction is based on *cryptographic accumulator*s [BDM93]. The idea is to add all elements of set $\mathcal{X}$ to the accumulator, sign the accumulator value with an existentially unforgeable signature scheme, and output a signature that consists of the accumulator value, a witness w_x for all $x \in \mathcal{X}$, and the signed accumulator value. To redact the signature, delete the associated witnesses. The construction is unforgeable, private, and transparent. They arrive at a vector construction by leveraging the basic idea in the set construction together with a tag to aid in preserving relative order of message blocks. In particular to sign the ith element, all tags for the $i-1$ blocks preceding are accumulated into an accumulator, and a signature on the tag, the element, and accumulator value is generated and added along with the ith witness to the output. Redaction consists of deleting accumulators, tags, witnesses, and associated signatures.

More recently, Sanders gave a RSS for sets from the Pointcheval-Sanders signature scheme [San20]. This scheme requires type-3 bilinear pairings $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$[2]. Of particular interest, this construction allows the key generator to limit which parties can redact messages; as in order to redact, special elements of $\mathbb{G}_1$ must be possessed. However, the size $|S|$ of the set S must be known at the time of key generation as the signing and verification keys depend on this size. The size of the signing key is $\Theta(|S|)$ and the size of the verification key is $\Theta\left(|S|^2\right)$. The scheme of Sanders is not the only unlinkable scheme, Camenisch et. al achieved unlinkablity through the use of NIZKs in [CDHK15] using bilinear groups. Additionally, [CDHK15] obtained UC secure redactable signatures.

The notion of redactable signature schemes have been cast into a general framework by Ahn et. al through the notion of P-Homomorphic signatures [ABC+15] as well as by Derler et. al [DPSS15]. While prior works tied to a specific class of data structure (vectors, sets, etc.) and security notions, Derler et. al focused on generic definitions that unify the space. Ahn et. al refer to their redactable signature schemes as *quoting* schemes, and are merely a type of P-Homomorphic signatures.

Some work has been undertaken related to the notion of redaction control, called *controlled disclosure* [MIM+05, HHH+08]. In controlled disclosure the *redactor* is allowed to declare certain regions of a document

[1] [JMSW02] offers slightly different trivial construction that only offers existential unforgeability under redaction.

[2] There is no efficiently computable homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

as unredactable. This declaration is made during initial redaction. The signer has no control over what is and is not marked as redactable. The construction of [MIM+05] is based on the content extraction signature schemes of [SBZ01]. The work of [HHH+08] is based on the vector construction of Johnson et. al.

2 Definitions

2.1 Notation

We denote by $\mathsf{negl}(\cdot)$ an arbitrary negligible function. We write $x \xleftarrow{\$} A$ to denote that x is sampled from set A uniformly at random. Denote by $x \parallel y$ the concatenation of two binary strings x and y. We denote by (G, S, V) an existentially unforgeable under chosen message attack (EUF-CMA) secure digital signature scheme. The signing key will be denoted by sk and the verification key will be denoted vk. We denote by $\mathsf{Merkle}(A)$ the root of the Merkle tree on set A. Moreover for $a \in A$, let $\mathsf{Proof}(a, v)$ be the proof that element a is in a Merkle tree with root v. Denote by $\mathsf{CheckProof}(a, \Pi, v)$ the result of checking proof Π for the statement that a is in the Merkle tree represented by root v.

Let $\mathcal{P}(\mathcal{U})$ denote the powerset of a set $\mathcal{U}$. A policy P is a subset of the powerset of the universe $\mathcal{U}$ (i.e., $P \subseteq \mathcal{P}(\mathcal{U})$). When discussing policies we will often abuse functional notation. In particular, $P(S) = 1$ denotes $S \in P$ and $P(S) = 0$ denotes $S \notin P$. We denote the natural number corresponding to the characteristic bitstring of set A relative to universe $\mathcal{U}$ by $\mathcal{X}_{\mathcal{U}}(A)$. Finally, we let $[n]$ denote the set $\{1, 2, \ldots, n\}$.

2.2 Constrained Pseudorandom Functions

Our use of constrained pseudorandom functions will be based on inner-product predicates thus, we will tailor our definition to this case.

A constrained pseudorandom function (CPRF) for inner-product predicates is a tuple of algorithms $\Pi = (\mathsf{F.Gen}, \mathsf{F.Constrain}, \mathsf{F.Eval}, \mathsf{F.ConstrainedEval})$ defined over domain $\mathcal{X}$ of vectors of some fixed dimension and range $\mathcal{Y}$ with the following properties:

- $\mathsf{F.Gen}\left(1^\lambda, \ell\right)$ takes as input a security parameter, 1^λ and samples a random function from PRF family with vectors of size ℓ by sampling a master key msk from the appropriate keyspace.
- $\mathsf{F.Constrain}(\mathsf{msk}, \vec{z})$ takes the master key and vector $\vec{z}$ as input and returns a key $k_{\vec{z}}$ that represents the constrained key that can evaluate the CPRF on any value $\vec{x}$ such that $\langle \vec{z}, \vec{x} \rangle = 0$.
- $\mathsf{F.Eval}(\mathsf{msk}, \vec{x})$ takes the master key and an input $\vec{x}$ of dimension q and returns the appropriate value $y \in \mathcal{Y}$.
- $\mathsf{F.ConstrainedEval}(k_{\vec{z}}, \vec{x})$ takes as input a constrained key $k_{\vec{z}}$ and an $\vec{x} \in \mathcal{X}$. If $\langle \vec{z}, \vec{x} \rangle = 0$ the correct value $y \in \mathcal{Y}$ is returned (i.e. $\mathsf{Eval}(k_{\vec{z}}, \vec{x})$); otherwise either a random value in $\mathcal{Y}$ or $\perp$ is returned.

Correctness We say a CPRF is *correct* if the evaluation of $\vec{x}$ using the constrained key agrees with the evaluation of $\vec{x}$ using the master key only when the predicate associated with the constrained key is satisfied. More formally, F is *correct* if for all $\vec{x} \in \mathcal{X}$ where $\langle \vec{z}, \vec{x} \rangle = 0$, master keys msk, and constrained keys $k_{\vec{z}}$, $\mathsf{F.Eval}(\mathsf{msk}, x) = \mathsf{F.ConstrainedEval}(k_{\vec{z}}, x)$.

Security Notion A constrained pseudorandom function has a nuanced security guarantee. With a traditional PRF, security states that under a polynomial number of adaptive queries a probabilistic-polynomial time (PPT) adversary can not distinguish between the output of a PRF and the output of a randomly selected function of the same domain and range. When it comes to inner-product predicate CPRFs security is relative to the inner product where the adversary will be able to evaluate the PRF at any vector $\vec{x} \in \mathcal{X}$ where the inner product with $\vec{x}$ is zero, without the assistance of an evaluation oracle. The formalized security notion is in figure 1.

$\mathsf{Exp}_{\mathcal{A}}^{\text{CPRF-Security}}\left(1^{\lambda}\right)$

1 : $\mathsf{msk} \leftarrow \mathsf{F.Gen}\left(1^{\lambda}, \ell\right)$
2 : $\vec{z} \leftarrow \mathcal{A}^{\mathcal{O}_1}\left(1^{n}\right)$
3 : $k_{\vec{z}} \leftarrow \mathsf{F.Constrain}\left(\mathsf{msk}, \vec{z}\right)$
4 : $\vec{x}^{*} \leftarrow \mathcal{A}^{\mathcal{O}_1}\left(1^{n}, k_{\vec{z}}\right)$
5 : **if** $\vec{x}^{*} \in Q \vee \langle \vec{x}, \vec{z} \rangle = 0$, **abort**
6 : $b \xleftarrow{\$} \{0,1\}$
7 : **if** $b = 0$ **then**
8 : $\quad y \leftarrow \mathsf{F.Eval}\left(\mathsf{msk}, \vec{x}^{*}\right)$
9 : **else**
10 : $\quad y \xleftarrow{\$} \mathcal{Y}$
11 : $b' \leftarrow \mathcal{A}^{\mathcal{O}_2}\left(1^{n}, y\right)$
12 : **return** $b' = b$

Oracle $\mathcal{O}_1\left(\vec{x}\right)$

1 : $Q = Q \cup \{\vec{x}\}$
2 : **return** $\mathsf{F.Eval}\left(\mathsf{msk}, \vec{x}\right)$

Oracle $\mathcal{O}_2\left(\vec{x}\right)$

1 : **if** $\vec{x} = \vec{x}^{*}$, **abort**
2 : **return** $\mathsf{F.Eval}\left(\mathsf{msk}, \vec{x}\right)$

Figure 1: The CPRF security game, our specific form is 1-key adaptive security focused on inner-product predicates.

In this work we make use of the construction of [SS25]. For clarity of exposition we give the simplest form in the random oracle model. The idea is that the CPRF key is a random vector $\vec{z}_0$ over some field $\mathcal{F}$. To evaluate the CPRF on a vector $\vec{x}$, the inner product of $\vec{x}$ and $\vec{z}_0$ is computed this results in a key k. The result of evaluation is given by $H\left(k, \vec{x}\right)$ where H is a random oracle. The constrain algorithm takes a vector $\vec{z}$ and constructs a constrained key such that evaluation is correct only when the inner product of $\vec{z}$ and $\vec{x}$ is zero. The constrained key $\vec{z}_1$ is given by $\vec{z}_1 = \vec{z}_0 + \Delta\vec{z}$ where Δ is some random element from field $\mathcal{F}$. The constrained evaluation is identical to the unconstrained evaluation. The complete construction is given in figure 2. The construction can be modified slightly to establish security in the standard model, using an related-key attack secure PRF in place of the random oracle [SS25].

$\mathsf{F.Gen}\left(1^{\lambda}, \ell\right)$

1 : $\vec{z}_0 \xleftarrow{\$} \mathbb{F}^{\ell}$
2 : **return** $\vec{z}_0$

$\mathsf{F.Constrain}\left(\vec{z}_0, \vec{z}\right)$

1 : $\Delta \xleftarrow{\$} \mathbb{F}$
2 : $\vec{z}_1 \leftarrow \vec{z}_0 - \Delta\vec{z}$
3 : **return** $\vec{z}_1$

$\mathsf{F.Eval}\left(\vec{z}_0, \vec{x}\right)$

1 : $k \leftarrow \langle \vec{z}_0, \vec{x} \rangle$
2 : **return** $H\left(k, \vec{x}\right)$

$\mathsf{F.CEval}\left(\vec{z}_1, \vec{x}\right)$

1 : $k \leftarrow \langle \vec{z}_1, \vec{x} \rangle$
2 : **return** $H\left(k, \vec{x}\right)$

Figure 2: Inner-product CPRF using random oracle H due to [SS25]

2.3 Commitment

A commitment scheme is a tuple of polynomial algorithms $\Pi = (\mathsf{Commit}, \mathsf{Open})$ where,

- $\mathsf{Commit}\left(m, r\right)$: Give message m and randomness r output a commitment c.
- $\mathsf{Open}\left(c, m, r\right)$: Given message m, commitment c, and randomness r, output 1 if c is a commitment to m and 0 otherwise.

Correctness A commitment scheme is correct if for all possible messages m and randomness r, $\mathsf{Open}\left(\mathsf{Commit}\left(m,r\right),m,r\right)=1$.

Security There are two main security properties for a commitment scheme, binding and hiding. Intuitively, binding says that a commitment cannot be opened to two different messages. Formally, we say a commitment scheme is binding if for a PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\text{Binding}}\left(1^{\lambda}\right)=1\right]\leq\mathsf{negl}\left(\lambda\right)$. Hiding says that from the commitment, nothing can be learned about the committed message. We will say that a commitment scheme is hiding if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\text{Hiding}}\left(1^{\lambda}\right)=1\right]\leq\frac{1}{2}+\mathsf{negl}\left(\lambda\right)$. The security games are in figure 3

$\mathsf{Exp}_{\mathcal{A}}^{\text{Hiding}}\left(1^{\lambda}\right)$

1 : $(m_0, m_1) \leftarrow \mathcal{A}\left(1^{\lambda}\right)$
2 : $b \xleftarrow{\$} \{0,1\}$
3 : $r \xleftarrow{\$} \{0,1\}^*$
4 : $c \leftarrow \mathsf{Commit}\left(m_b, r\right)$
5 : $b' \leftarrow \mathcal{A}\left(1^{\lambda}, c\right)$
6 : **return** $b' = b$

$\mathsf{Exp}_{\mathcal{A}}^{\text{Binding}}\left(1^{\lambda}\right)$

1 : $(c, m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}\left(1^{\lambda}\right)$
2 : **return** $m_0 \neq m_1 \wedge \mathsf{Commit}\left(m_0, r_0\right) = \mathsf{Commit}\left(m_1, r_1\right)$

Figure 3: The security notions for a commitment scheme

In this work we will make use of the classic hash-based commitment scheme. The construction requires a hash function $h:\{0,1\}^{n}\rightarrow\{0,1\}^{m}$. The commitment c to a message m using randomness r is given by $c=h\left(m\parallel r\right)$. To verify that c is a commitment to message m with opening value r, verify that c is equal to $h\left(m\parallel r\right)$. This construction is binding based on the collision resistance property of the hash function. The construction is hiding if we treat the hash function as a random oracle. In this work we change how randomness is sampled, instead of using a uniformly random r in the commitment we utilize the output of a CPRF. The hiding and binding properties of the modified construction follow simply.

In this work we will further need a method for committing to a set of n elements $A=\{a_1,a_2,\ldots,a_n\}$. Given a random oracle $H:\{0,1\}^{*}\rightarrow\{0,1\}^{\ell}$, we define the commitment to a set A as $H\left(H\left(a_1\right)\parallel\cdots\parallel H\left(a_n\right)\parallel r\right)$. To verify the commitment, recompute the commitment and compare.

3 Redactable Set Signatures with Policies

A redactable set signature scheme is a signature scheme that allows a redactor, without possessing the signing key, to convert a signature on set S to a signature on set S' if $S'\subset S$. This paper introduces a *policy-based redactable set signature scheme*. These signatures allow for a signer to limit the possible future redactions by specifying a policy $P\subseteq\mathcal{P}\left(\mathcal{U}\right)$ as signing time. The policy ensures that a redactor can only redact a signature on set S to a signature on set S' if $S'\subseteq S$ *and* the policy P contains S' (i.e. $P\left(S'\right)=1$).

3.1 Syntax

Formally, a redactable signature for sets with a policy is a four tuple $(\mathsf{Gen},\mathsf{Sign},\mathsf{Redact},\mathsf{Vrfy})$ of polynomial time algorithms.

- $\mathsf{Gen}\left(1^{\lambda},\mathcal{U}\right)$: a probabilistic algorithm that takes the security parameter λ as input and the set universe $\mathcal{U}$ and returns a signing key sk, a verification key vk, and an injective function $\mathsf{ord}:\mathcal{U}\rightarrow\mathbb{N}$.
- $\mathsf{Sign}_{sk}\left(S,P\right)$ a possibly probabilistic algorithm takes a signing key sk and a set $S\subseteq\mathcal{U}$ as input and returns a signature σ on set S with redaction policy P.

- $\mathsf{Redact}_{vk}(S, \sigma, S')$ a possible probabilistic algorithm that takes the verification key vk, a set S, its signature σ with policy P, and and an S' such that $S' \subset S$ as input. The algorithm returns σ' if and only if $P(S') = 1$ and $\mathsf{Vrfy}_{vk}(S, \sigma) = 1$.
- $\mathsf{Vrfy}_{vk}(S, \sigma)$ a deterministic algorithm that takes the verification key vk, the set S , and purported signature on S, σ and returns 1 if σ is a valid signature for S and 0 otherwise.

3.2 Security Notions

There are three common security notions for redactable signatures found in the literature. These are unforgeability, privacy, and unlinkability. To these standard notions we add notions of policy privacy and single redaction. The definitions of the common notions have been adapted to policy-based set redaction schemes.

Unforgeability It should be hard for an adversary to forge a valid signature on a message that is *not* a redaction of a message they have already seen. It should be noted that this notion is based on a small modification to traditional existential unforgeability under a chosen message attack (EUF-CMA). Effectively only changing the condition on what it means to be a valid forgery. Since any party can construct a redaction, there is no redaction oracle. The formal security game is found in figure 4. We say an RSS is unforgeable if for all PPT adversaries $\mathcal{A}$ and all $n \in \mathbb{N}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{Forge}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] \leq \mathsf{negl}(\lambda)$.

Privacy From a redacted signature message pair, it should be infeasible to derive information on the parts of the message that have been redacted. It should be noted this does not preclude the adversary from learning what parts of the message were redacted. The formal security game is found in figure 4. We say that a redactable signature scheme is private if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{Privacy}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$.

Unlinkability It should be hard for an adversary to link a redacted signature back to the original signature, even for an adversary that generated the public key. The security game for unlinkability can be found in figure 4. We say that a redactable signature scheme is unlinkable if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{unlink}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] = \frac{1}{2} + \mathsf{negl}(\lambda)$.

Policy Privacy At times it may be important for a policy-based redactable set signature to have a notion of policy privacy. Informally, policy privacy means that no adversary can discern the policy associated with the signature or redacted signature. The formal definition of security is found in figure 5. We say a policy-based redactable set signature scheme is policy private if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{PolPriv}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$. If a scheme does not support policy privacy, we say that the scheme is a *public policy* scheme.

Policy privacy, as the game is specified may be too restrictive for some applications. Consider, for example, the case where a policy must only remain private for a party verifying a redacted signature. In this case we can offer a weaker form of policy privacy where the adversary must determine the policy from only seeing a redacted signature. The formal definition of security is found in figure 5. We say that a policy-based redactable set signature scheme is *weakly policy private* if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{PolPriv}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$.

Redact Once In some applications it may be important for a redacted signature to not allow further redactions. We call such signatures *redact once*. Specifically given a redacted signature σ for set S, it is computationally infeasible for the adversary to produces a signature σ^* for set $S^* \subset S$ even if $P(S^*) = 1$. Se say a policy-based redactable set signature scheme is redact once if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{SingleRedact}}\left(1^{\lambda}, \mathcal{U}\right) = 1\right] \leq \mathsf{negl}(\lambda)$

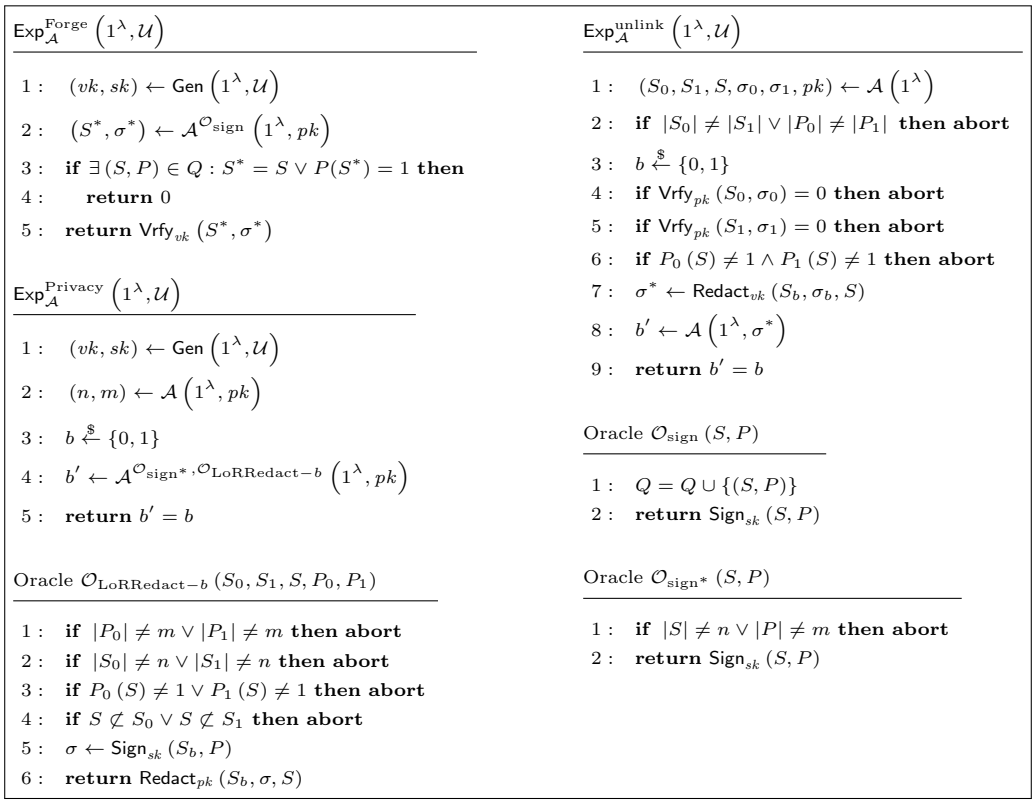


Figure 4: Unforgeability, privacy, transparency, and unlinkability security notions. The predicates P_0 and P_1 are policies for their respective signatures.

4 Public Policy Scheme

Our goal is to provide a redactable signature scheme that embeds a policy into the signature scheme such that only certain redactions are allowed. Specifically our construction allows the signer to limit the which subsets of the signed set can arise via redaction.

Given a constrained PRF for inner-product constraints ($\mathsf{F.Gen}, \mathsf{F.Constrain}, \mathsf{F.Eval}, \mathsf{F.CEval}$) from section 2.2, a commitment scheme ($\mathsf{Commit}, \mathsf{Open}$) for sets from section 2.3, and an EUF-CMA secure digital signature algorithm (G, S, V) we can construct a policy based redactable set signature scheme RSSS = ($\mathsf{Gen}, \mathsf{Sign}, \mathsf{Redact}, \mathsf{Verify}$). Our scheme is summarized in figure 6 for a sets from universe $\mathcal{U}$ and a policy $P \subseteq \mathcal{P}(\mathcal{U})$.

Construction Overview To construct a signature for a set S with policy P, we first commit to S and all subsets in policy P using a the commitment scheme. We denote this set of commitments by $\mathcal{C}$. We proceed to determine the root of the Merkle tree constructed from $\mathcal{C}$, denoted $\mathsf{Merke}(\mathcal{C})$. Finally, we sign the Merkle root. The signatures on set S is $(\mathsf{S}_{sk}(\mathsf{Merkle}(\mathcal{C})), \mathcal{C}, P, k)$, where k is the CPRF key that represents a the vector of coefficients of a polynomial whose roots are $\{\mathcal{X}_{\mathcal{U}}(A) \mid A \in P\}$.

We redact the signature $(\sigma_v, \mathcal{C}, P, k)$ on set S to a signature on S' by outputting the signature $(\sigma_v, \mathcal{C}, P', k')$ where $P' = \{A \in P \mid A \subseteq S'\}$ and k' is the CPRF key that represent the vector of coefficients of a polynomial whose roots are $\{\mathcal{X}_{\mathcal{U}}(A) \mid A \in P'\}$.

To verify a signature $(\sigma_v, \mathcal{C}, P, k)$ on set S, we must verify three facts. The first is that that $P(S) = 1$. The second, $V(\mathsf{Merkle}(\mathcal{C}), \sigma_v) = 1$. Lastly, we must verify that $\mathsf{Commit}(S, \mathsf{F.Eval}(k, \mathsf{Powers}(\mathcal{X}_{\mathcal{U}}(S)))) \in \mathcal{C}$. If all conditions verify, the signature is valid; otherwise, the signature is invalid.

$\mathsf{Exp}^{\mathrm{PolPriv}}_{\mathcal{A}}\left(1^{\lambda},\mathcal{U}\right)$

$(vk, sk) \leftarrow \mathsf{Gen}\left(1^{\lambda},\mathcal{U}\right)$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{sign}-b}}\left(1^{\lambda}, pk\right)$
return $b' = b$

$\mathsf{Exp}^{\mathrm{WeakPolPriv}}_{\mathcal{A}}\left(1^{\lambda},\mathcal{U}\right)$

$(vk, sk) \leftarrow \mathsf{Gen}\left(1^{\lambda},\mathcal{U}\right)$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{redact}-b}}\left(1^{\lambda}, pk\right)$
return $b' = b$

$\mathsf{Exp}^{\mathrm{SingleRedact}}_{\mathcal{A}}\left(1^{\lambda},\mathcal{U}\right)$

$(vk, sk) \leftarrow \mathsf{Gen}\left(1^{\lambda},\mathcal{U}\right)$
$\left(\left(S^*, \sigma^*\right), (S, \sigma)\right) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{redact}}}\left(1^{\lambda}, pk\right)$
return $\left(S^*, \sigma^*\right) \notin Q \wedge \mathsf{Vrfy}_{vk}\left(S^*, \sigma^*\right) = 1 \wedge S^* \subset S$

Oracle $\mathcal{O}_{\mathrm{sig}-b}\left(S, P_0, P_1\right)$

if $|P_0| \neq |P_1|$ **then abort**
if $P_0 = P_1$ **then abort**
if $P_0(S) \neq 1 \vee P_1(S) \neq 1$ **then abort**
return $\mathsf{Sign}_{sk}(S, P_b)$

Oracle $\mathcal{O}_{\mathrm{redact}}\left(S, S', P\right)$

if $S' \not\subset S \vee P\left(S'\right) \neq 1$ **then abort**
$\sigma = \mathsf{Sign}_{sk}(S, P)$
$\sigma' = \mathsf{Redact}_{vk}\left(S, \sigma, S'\right)$
$Q = Q \cup \left\{\left(S', \sigma'\right)\right\}$
return σ'

Oracle $\mathcal{O}_{\mathrm{redact}-b}\left(S_0, S_1, P_0, P_1, S'\right)$

if $S' \not\subset S_0 \vee S' \not\subset S_1$ **then abort**
if $P_0\left(S'\right) \neq 1 \vee P_1\left(S'\right) \neq 1$ **then abort**
if $|P_0| \neq |P_1|$ **then abort**
$\sigma = \mathsf{Sign}_{sk}(S_b, P_b)$
$\sigma' = \mathsf{Redact}_{vk}\left(S_b, \sigma, S'\right)$
return σ'

Figure 5: Advanced security notions for policy-based redactable set signatures.

$\mathsf{Gen}\left(1^{\lambda},\mathcal{U}\right)$

$(vk, sk) \leftarrow \mathsf{G}\left(1^{\lambda}\right)$
return $(vk, sk, \mathsf{ord} : \mathcal{U} \to \mathbb{N})$

$\mathsf{Vrfy}_{vk}(S, \sigma)$

parse $\sigma = (\sigma_v, \mathcal{C}, P, k)$
if $P(S) \neq 1$ **then return** 0
if $\mathsf{V}_{vk}\left(\sigma_v, \mathsf{Merkle}(\mathcal{C})\right) \neq 1$ **then return** 0
return $\mathsf{Commit}\left(S, \mathsf{F.Eval}\left(k, \mathsf{Powers}\left(\mathcal{X}_{\mathcal{U}}(S)\right)\right)\right) \in \mathcal{C}$

$\mathsf{Sign}_{sk}(S, P)$

$\mathsf{msk} \leftarrow \mathsf{F.Gen}\left(1^n, |\mathcal{U}|\right)$
$\mathcal{C} \leftarrow \{\}$
$R \leftarrow \{\}$
for $S' \in P \cup \{S\}$ **do**
  $\mathcal{C} \leftarrow \mathcal{C} \cup \left\{\mathsf{Commit}\left(S', \mathsf{F.Eval}\left(\mathsf{msk}, \mathsf{Powers}\left(\mathcal{X}_{\mathcal{U}}\left(S'\right)\right)\right)\right)\right\}$
  $R \leftarrow R \cup \left\{\mathcal{X}_{\mathcal{U}}\left(S'\right)\right\}$
return $\left(\mathsf{S}_{sk}\left(\mathsf{Merkle}(\mathcal{C})\right), \mathcal{C}, P, \mathsf{F.Constrain}\left(\mathsf{msk}, \mathsf{toVec}(\mathsf{R})\right)\right)$

$\mathsf{Redact}_{vk}\left(S, \sigma, S'\right)$

parse $\sigma = (\sigma_v, \mathcal{C}, P, k)$
if $S' \not\subset S \vee P\left(S'\right) \neq 1$ **then abort**
if $\mathsf{Commit}\left(S', \mathsf{F.Eval}\left(k, \mathcal{X}_{\mathcal{U}}\left(S'\right)\right)\right) \notin \mathcal{C}$ **then abort**
$P' \leftarrow \{\}$
for $A \in P$ **do**
  if $A \subseteq S'$ **then**
    $R \leftarrow R \cup \{\mathcal{X}_{\mathcal{U}}(A)\}$
    $P' \leftarrow P' \cup \{A\}$
return $\left(\sigma_v, \mathcal{C}, P', \mathsf{F.Constrain}\left(k, \mathsf{toVec}(R)\right)\right)$

Figure 6: The policy-based set redactable signature scheme. The $\mathsf{toVec}(R)$ functions builds the associated vector of coefficients of the polynomial whose roots are R. The $\mathsf{powers}(x)$ function builds a vector of the powers of x the same size as the CPRF key.

4.1 Security Analysis

Theorem 1. *If* (G, S, V) *is an EUF-CMA secure signature scheme,* $(\mathsf{Commit}, \mathsf{Open})$ *is a commitment scheme, and a Merkle tree is collision resistant, then RSSS is unforgeable.*

Proof. We sketch a proof of our theorem. Let $\mathcal{A}$ be an adversary that wins the unforgeability game with RSSS. We can use $\mathcal{A}$ to construct an adversary $\mathcal{B}_0$ that can construct a forgery for signature scheme (G, S, V). Let's assume that $\mathcal{A}$ wins the unforgeability game by constructing a fresh signature $(\sigma^*, \mathcal{C}^*, k^*)$ where the binding property of the commitment scheme was not violated nor is the collision resistance property of the Merkle tree. Since $\mathcal{B}_0$ is utilizing adversary $\mathcal{A}$ it must simulate the signing oracle for $\mathcal{A}$. This is done by following the signing algorithm of RSS. When the signing algorithm of RSS must sign the Merkle root, $\mathcal{B}_0$ will query its signing oracle on the Merkle root. When $\mathcal{A}$ outputs its forgery $(\sigma^*, \mathcal{C}^*, k^*)$, $\mathcal{B}_0$ outputs $(\sigma^*, \mathsf{Merkle}(\mathcal{C}^*))$.

Using $\mathcal{A}$ we can construct an adversary $\mathcal{B}_1$ that can violate the binding property of the commitment scheme $(\mathsf{Commit}, \mathsf{Open})$. Let's assume that $\mathcal{A}$ wins the unforgeability game by constructing a fresh signature $(\sigma^*, \mathcal{C}^*, k^*)$ where neither the EUF-CMA property of the (G, S, V) nor the collision resistance property of the Merkle tree is violated. Since $\mathcal{B}_1$ is utilizing adversary $\mathcal{A}$ it must simulate the signing oracle for $\mathcal{A}$. This is done by following the signing algorithm of RSS. When calculating the ith individual commitments, $\mathcal{B}_1$ saves the triple (S_i, c_i, r_i) associated with its corresponding redactable signature. When $\mathcal{A}$ outputs its forgery $(\sigma^*, \mathcal{C}^*, k^*)$, based on our assumptions, there must be at least one commitment $c^* \in \mathcal{C}^*$ that matches a c' in a signature $(\sigma', \mathcal{C}', k')$ where $\sigma' = \sigma^*$ but $S' \neq S^*$. In this case, $\mathcal{B}_1$ outputs the pairs $(S', c', \mathsf{F.Eval}(k', \mathsf{Powers}(\mathcal{X}_{\mathcal{U}}(S'))))$ and $(S^*, c^*, \mathsf{F.Eval}(k^*, \mathsf{Powers}(\mathcal{X}_{\mathcal{U}}(S^*))))$.

Using $\mathcal{A}$ we can construct an adversary $\mathcal{B}_2$ that can violate the collision resistance property of the Merkle tree. Let's assume that $\mathcal{A}$ wins the unforgeability game by constructing a fresh signature $(\sigma^*, \mathcal{C}^*, k^*)$ where neither the EUF-CMA property of the (G, S, V) nor the binding property of the underlying commitment scheme is violated. Since $\mathcal{B}_2$ is utilizing adversary $\mathcal{A}$ it must simulate the signing oracle for $\mathcal{A}$. Adversary $\mathcal{B}_2$ does this by following the RSS algorithm exactly except that the Merkle root, commitment set are stored indexed by the signature. When $\mathcal{A}$ outputs its forgery $(\sigma^*, \mathcal{C}^*, k^*)$, based on our assumptions, there must be some signature $(\sigma', \mathcal{C}', k')$ such that $\mathsf{Merkle}(\mathcal{C}*) = \mathsf{Merkle}(\mathcal{C}')$. Adversary $\mathcal{B}_2$ outputs the colliding pair $(\mathcal{C}^*, \mathcal{C}')$. □

Intuitively, since the signatures are generated using a commitment scheme the redacted components remain hidden in a signature

Theorem 2. $(\mathsf{Commit}, \mathsf{Open})$ *is a commitment scheme and* $(\mathsf{F.Gen}, \mathsf{F.Constrain}, \mathsf{F.Eval}, \mathsf{F.ConstrainedEval})$ *is a secure CPRF, then RSSS is private.*

Proof. We sketch a proof of our theorem. Let $\mathcal{A}$ be an adversary that wins the privacy game with RSSS. We can use $\mathcal{A}$ to build an adversary $\mathcal{B}$ that wins the hiding game. We will assume that $\mathcal{A}$ makes at most q left-or-right oracle queries. $\mathcal{B}$ begins by running $\mathsf{Gen}\left(1^\lambda, \mathcal{U}\right)$ to obtain (vk, sk). Furthermore, $\mathcal{B}$ will randomly select $i \in [q]$ and $b \in \{0, 1\}$. To simulate $\mathcal{O}_{\mathrm{sign}^*}$, $\mathcal{B}$ will directly follow the description of the oracle from the security game, as $\mathcal{B}$ posses the signing key necessary to construct the signature. To simulate the $\mathcal{O}_{\mathrm{LorRRedact}-b}$ oracle $\mathcal{B}$ follows the process described in the oracle definition. Since $\mathcal{B}$ possess the signing key, it can follow the process of the signature scheme exactly. However, on the ith query we submit a pair of subsets $A \subset S_0$ and $B \subset S_1$ in P_b. If $A \subseteq S$ or $B \subseteq S$, abort. If we do not abort the result of $\mathcal{B}$'s game is substituted as the appropriate commitment. Once completed, the signature is redacted to S and submitted to $\mathcal{A}$. Eventually, $\mathcal{A}$ will output a bit $b' \in \{0, 1\}$. If $b = b'$, $\mathcal{B}$ should output b'. Otherwise, $\mathcal{B}$ should output a random bit. We assumed that $\mathcal{A}$ can win privacy game with better than $\frac{1}{2} + \mathsf{negl}(\lambda)$ probability, thus the probability that $\mathcal{B}$ wins the game is greater than $\frac{1}{2} + \frac{1}{2q} + \frac{\mathsf{negl}(\lambda)}{q}$. □

4.2 Extensions

Our construction can be made redact once by removing the policy from the signature and constraining the CPRF key to only evaluate $\mathcal{X}_{\mathcal{U}}(S')$. The full construction, we call RO − RSS is in figure 7.

$\mathsf{Gen}\left(1^{\lambda}, \mathcal{U}\right)$

1 : $(vk, sk) \leftarrow \mathsf{G}\left(1^{\lambda}\right)$
2 : **return** $(vk, sk, \mathsf{ord} : \mathcal{U} \to \mathbb{N})$

$\mathsf{Vrfy}_{vk}\left(S, \sigma\right)$

1 : **parse** $\sigma = (\sigma_v, \mathcal{C}, k)$
2 : **if** $\mathsf{V}_{vk}\left(\sigma_v, \mathsf{Merkle}\left(\mathcal{C}\right)\right) \neq 1$ **then return** 0
3 : **return** $\mathsf{Commit}\left(S, \mathsf{F.Eval}\left(k, \mathsf{Powers}\left(\mathcal{X}_{\mathcal{U}}\left(S\right)\right)\right)\right) \in \mathcal{C}$

$\mathsf{Sign}_{sk}\left(S, P\right)$

1 : $\mathsf{msk} \leftarrow \mathsf{F.Gen}\left(1^{n}, |\mathcal{U}|\right)$
2 : $\mathcal{C} \leftarrow \{\}$
3 : $R \leftarrow \{\}$
4 : **for** $S' \in P \cup \{S\}$ **do**
5 : $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \left\{\mathsf{Commit}\left(S', \mathsf{F.Eval}\left(\mathsf{msk}, \mathsf{Powers}\left(\mathcal{X}_{\mathcal{U}}\left(S'\right)\right)\right)\right)\right\}$
6 : $\quad R \leftarrow R \cup \left\{\mathcal{X}_{\mathcal{U}}\left(S'\right)\right\}$
7 : **return** $\left(\mathsf{S}_{sk}\left(\mathsf{Merkle}\left(\mathcal{C}\right)\right), \mathcal{C}, \mathsf{F.Constrain}\left(\mathsf{msk}, \mathsf{toVec}\left(\mathsf{R}\right)\right)\right)$

$\mathsf{Redact}_{vk}\left(S, \sigma, S'\right)$

1 : **parse** $\sigma = (\sigma_v, \mathcal{C}, P, k)$
2 : **if** $S' \not\subset S$ **then abort**
3 : **if** $\mathsf{Commit}\left(S', \mathsf{F.Eval}\left(k, \mathsf{Powers}\left(\mathcal{X}_{\mathcal{U}}\left(S'\right)\right)\right)\right) \notin \mathcal{C}$ **then abort**
4 : **return** $\left(\sigma_v, \mathcal{C}, \mathsf{F.Constrain}\left(k, \mathsf{toVec}\left(\mathcal{X}_{\mathcal{U}}\left(S'\right)\right)\right)\right)$

Figure 7: The policy-based redact once set redactable signature scheme. The $\mathsf{toVec}\left(R\right)$ functions builds the associated vector of coefficients of the polynomial whose roots are R. The $\mathsf{powers}\left(x\right)$ function builds a vector of the powers of x the same size as the CPRF key.

To see that this is a redact once scheme observe that given a redacted signature $(\sigma, \mathcal{C}, k)$ for set S, the only commitment that k allows a party to recompute is the one that is associated with $\mathcal{X}_{\mathcal{U}}\left(S\right)$. This follows directly from the correctness property of the underlying CPRF. Additionally the scheme is unforgeable and private. The proofs mirror the proofs of the public-policy, thus have been elided.

We can provide a weakly policy private scheme by modifying how the redacted signatures are constructed in the redact once scheme. Instead of including the commitments to each set in the policy in the redact signature, we include a Merkle proof that subset is included in a Merkle tree with root v. Since the verification is now dependend on the type of signature, this scheme differentiates between original and redacted signatures. This is indicated in the signature by including the Merkle root in the redacted signature and not in the original signature. Our modified scheme is found in figure 8.

To see that this construction is weak policy private, observe that from the Merkle root, the proof, and the correctness of the CPRF no PPT adversary can determine any information about the other subsets contained in the policy.

4.3 Limits of Approach

There are several limits present in our candidate constructions. We first observe that none of our constructions are unlinkable. There are two reasons for this. The first reason is that the set of commitments in the signature are shared with the original signature and thus allow one to link back to the original signature. Secondly, the signature on the Merkle root allows an adversary to link back to the original signature.

Secondly, we observe that our extend redact once construction (figure 7) while being redact-once is not policy private, despite not carrying along the policy explicitly. An adversary given a signature can simply unitlize knowledge of the set S and utilize the CPRF key to check all possible subsets against the provided commitments. At first blush this appears to be the description of a non-PPT adversary, however, the number of subsets of S is not guaranteed to exceed some polynomial function of the security parameter. Thus an efficient adversary could violate the privacy for any policy whose size is polynomial in the security parameter.

Our redact-once weak policy private construction (figure 8) can achieve weak policy privacy, however, the stronger notion remains out of reach. The reason is similar to the redact once construction, the commitments provided in the signature allow for discovering the policy using the same testing procedure described earlier.

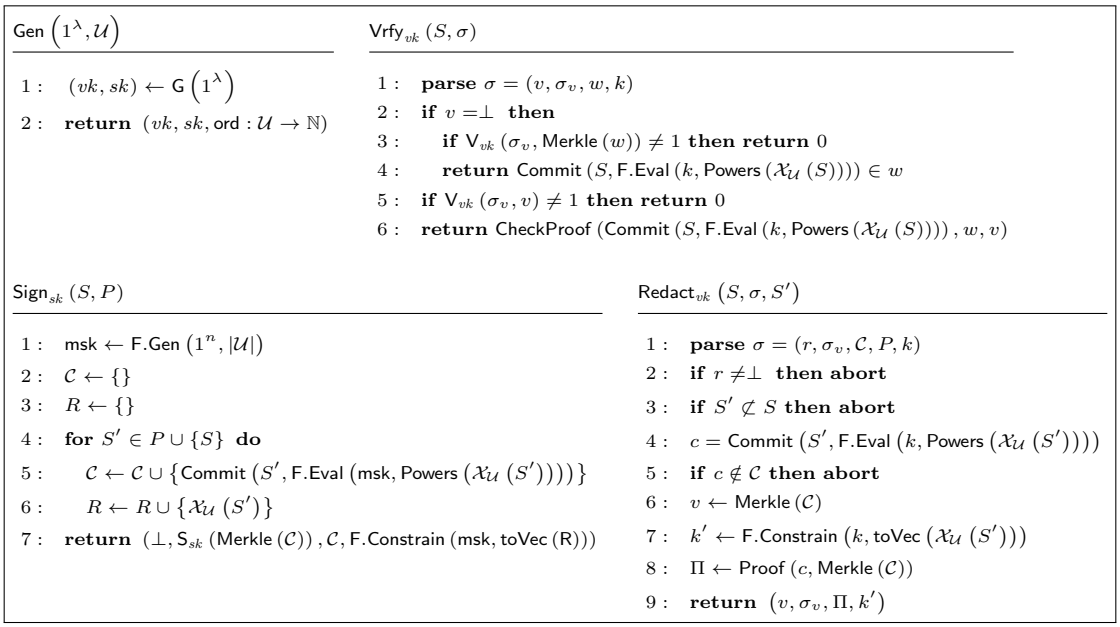


Figure 8: The policy-based redact once set redactable signature scheme with weak policy privacy. The $\mathsf{toVec}(R)$ functions builds the associated vector of coefficients of the polynomial whose roots are R. The $\mathsf{powers}(x)$ function builds a vector of the powers of x the same size as the CPRF key.

5 Conclusion

In this work we introduced policy-based redactable signature schemes and gave several practical constructions. We have shown a public policy construction, a redact once construction, and a redact once weak policy private construction.

While we have achieved our goals of providing a policy-based redactable signature scheme there are many future open directions. One of the most important is realizing an unlinkable scheme which would immediately provide an anonymous credential system. Another is reducing the size of the signatures. The current signatures have size linear in the number of sets in the policy. Finally, realizing a policy private construction is also left as an open question.

References

[ABC+15] Jae Hyun Ahn, Dan Boneh, Jan Camenisch, Susan Hohenberger, Abhi Shelat, and Brent Waters. Computing on authenticated data. *Journal of Cryptology*, 28(2):351–395, 2015.

[BBD+10] Christina Brzuska, Heike Busch, Oezguer Dagdelen, Marc Fischlin, Martin Franz, Stefan Katzenbeisser, Mark Manulis, Cristina Onete, Andreas Peter, Bertram Poettering, et al. Redactable signatures for tree-structured data: Definitions and constructions. In *International Conference on Applied Cryptography and Network Security*, pages 87–104. Springer, 2010.

[BDM93] Josh Benaloh and Michael De Mare. One-way accumulators: A decentralized alternative to digital signatures. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 274–285. Springer, 1993.

[BPS17] Arne Bilzhause, Henrich C Pöhls, and Kai Samelin. Position paper: the past, present, and future of sanitizable and redactable signatures. In *Proceedings of the 12th International Conference on Availability, Reliability and Security*, pages 1–9, 2017.

[CDHK15] Jan Camenisch, Maria Dubovitskaya, Kristiyan Haralambiev, and Markulf Kohlweiss. Composable and modular anonymous credentials: Definitions and practical constructions. In *Advances in Cryptology–ASIACRYPT 2015: 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29–December 3, 2015, Proceedings, Part II 21*, pages 262–288. Springer, 2015.

[Cha85] David Chaum. Showing credentials without identification: Signatures transferred between unconditionally unlinkable pseudonyms. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 241–244. Springer, 1985.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *International conference on the theory and applications of cryptographic techniques*, pages 93–118. Springer, 2001.

[CL02] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *International conference on security in communication networks*, pages 268–289. Springer, 2002.

[CLX09] Ee-Chien Chang, Chee Liang Lim, and Jia Xu. Short redactable signatures using random trees. In *Cryptographers' Track at the RSA Conference*, pages 133–147. Springer, 2009.

[DPSS15] David Derler, Henrich C Pöhls, Kai Samelin, and Daniel Slamanig. A general framework for redactable signatures and new constructions. In *ICISC 2015*, pages 3–19. Springer, 2015.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[HHH+08] Stuart Haber, Yasuo Hatano, Yoshinori Honda, William Horne, Kunihiko Miyazaki, Tomas Sander, Satoru Tezoku, and Danfeng Yao. Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, pages 353–362, 2008.

[JMSW02] Robert Johnson, David Molnar, Dawn Song, and David Wagner. Homomorphic signature schemes. In *Cryptographers' track at the RSA conference*, pages 244–262. Springer, 2002.

[MIM+05] Kunihiko Miyazaki, Mitsuru Iwamura, Tsutomu Matsumoto, Ryôichi Sasaki, Hiroshi Yoshiura, Satoru Tezuka, and Hideki Imai. Digitally signed document sanitizing scheme with disclosure condition control. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 88(1):239–246, 2005.

[San20] Olivier Sanders. Efficient redactable signature and application to anonymous credentials. In *IACR International Conference on Public-Key Cryptography*, pages 628–656. Springer, 2020.

[SBZ01] Ron Steinfeld, Laurence Bull, and Yuliang Zheng. Content extraction signatures. In *International Conference on Information Security and Cryptology*, pages 285–304. Springer, 2001.

[SS25] Sacha Servan-Schreiber. Constrained pseudorandom functions for inner-product predicates from weaker assumptions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 232–265. Springer, 2025.