Linking-Based Revocation for Group Signatures: A Pragmatic Approach for Efficient Revocation Checks

Daniel Slamanig, Raphael Spreitzer, and Thomas Unterluggauer

IAIK, Graz University of Technology, Graz, Austria
{daniel.slamanig|raphael.spreitzer|thomas.unterluggauer}@iaik.tugraz.at

Abstract. Group signature schemes (GSS) represent an important privacy-enhancing technology. However, their practical applicability is restricted due to inefficiencies of existing membership revocation mechanisms that often place a too large computational burden and communication overhead on the involved parties. Moreover, it seems that the general belief (or unwritten law) of avoiding online authorities by all means artificially and unnecessarily restricts the efficiency and practicality of revocation mechanisms in GSSs. While a mindset of preventing online authorities might have been appropriate more than 10 years ago, today the availability of highly reliable cloud computing infrastructures could be used to solve open challenges. More specifically, in order to overcome the inefficiencies of existing revocation mechanisms, we propose an alternative approach denoted as *linking-based revocation* (LBR) which is based on the concept of controllable linkability. The novelty of LBR is its transparency for signers and verifiers that spares additional computations as well as updates. We therefore introduce dedicated revocation authorities (RAs) that can be contacted for efficient (constant time) revocation checks. In order to protect these RAs and to reduce the trust in involved online authorities, we additionally introduce *distributed controllable linkability*. Using latter, RAs cooperate with multiple authorities to compute the required linking information, thus reducing the required trust. Besides efficiency, an appealing benefit of LBR is its generic applicability to pairing-based GSSs secure in the BSZ model as well as GSSs with controllable linkability. This includes the XSGS scheme, and the GSSs proposed by Hwang et al., one of which has been standardized in the recent ISO 20008-2 standard.

1 Introduction

Group signature schemes (GSSs) [17] represent an important privacy-enhancing technology. Such schemes allow users to anonymously prove affiliation to a managed group by issuing so called group signatures. Although such signatures do not reveal the identity of the users, in case of dispute a dedicated opening authority is able to identify a signer. GSSs are especially attractive in scenarios like public transport or subscription-based services, where there is no need for service providers to uniquely identify single users, but only to ensure that they are allowed to use the respective services. Nevertheless, service providers typically want to ensure that group membership can be efficiently revoked. However, membership revocation is a non-trivial task as (1) users are anonymous and their privacy should be protected even in case of revocation, and (2) the revocation of one user should not affect the signing capabilities of other users. Even though membership revocation has gained increasing attention in the last decade [2,8,9,23,24,38,44], existing mechanisms place a computational burden and communication overhead on signers and verifiers.

While it seems that existing concepts aim to prevent online authorities by all means, we show that a *paradigm shift* towards online authorities—which we will protect by means of threshold cryptography—allows for the most efficient (constant time), and most generic revocation mechanism for existing GSSs. Given a signature in question and a revocation list, a revocation authority determines the revocation status of a signer by using a dedicated trapdoor to (anonymously) link the signature against a list of revoked members. Hence, this authority still preserves the signer's anonymity as it is strictly less powerful than an opening authority who can determine the actual identity of the signer. Our somewhat *unconventional* proposal of using an *online* revocation authority overcomes many issues of existing revocation mechanisms and, thus, we believe that our revocation mechanism represents a valuable addendum to the portfolio of revocation mechanisms.

Online Requirement. Although our approach relies on an online authority for revocation checks, we argue that today many devices are already connected to the Internet permanently and rely on the availability of cloud computing infrastructures. The establishment of the Internet of Things (IoT) requires devices being connected to the Internet, either via WiFi or even embedded SIM cards, for various (sometimes dubious) reasons. Nevertheless, irrespective of whether existing IoT devices provide any useful features, the point is that many devices are already interconnected among each other and also extensively use Internet services based on cloud computing infrastructures. Thus, we consider an online RA as absolutely reasonable. In cases where network connectivity and availability are not an issue, our proposed revocation mechanism provides significant advantages compared to other revocation mechanisms. Further, since signature verification is decoupled from the online revocation checks, these revocation checks can also be postponed in case the revocation authority might not be available. Besides, as we will discuss later, we protect the required trapdoor information by means of threshold cryptography in order to reduce the risk of its exposure.

Contributions. The contributions of this work can be summarized as follows.

– We suggest a paradigm shift towards online revocation checks for group signatures by relying on online revocation authorities (RAs). Although currently this seems to be prevented by all means, it, however, allows us to come up with a privacy-respecting constant-time revocation mechanism that can be generically applied to a large class of pairing-based GSSs. In doing so, we use the concept of controllable linkability for group signatures.

– We introduce the concept of distributed controllable linkability. This is a threshold variant of controllable linkability, which requires a predefined number t of linking authorities to cooperate in order to link signatures. Using this concept, we can reduce the trust in and also improve the robustness of RAs. Besides, it may be of independent interest as a feature on its own.
– We demonstrate the ease of applicability of the linking-based revocation mechanism using the well-known XSGS group signature scheme [21].

2 State-of-the-Art in Revocation and Motivation

Below we discuss efficiency considerations of existing revocation mechanisms using the metric of additional computations and updates required for signers as well as verifiers. Subsequently, R denotes the number of revoked members and N the number of group members.

Basic Approaches. The most basic approach is *reissuance-based revocation* [1] which requires all non-revoked members to receive new group signing keys. Similarly, *credential-update revocation* (CUR) [7] requires non-revoked members to update their group signing key on every revocation. Both mechanisms suffer from additional communication and computation overhead in case of frequent revocations. In particular, $\mathcal{O}(R)$ (multi-)exponentiations for signers.

Blacklist Revocation. *Certificate-based blacklist revocation* (BR-C) [10] requires signers to provide a zero-knowledge proof that they are not listed on a revocation list (RL), which means that signers/verifiers need to perform $\mathcal{O}(R)$ computations for every sign/verify operation and the signature size also increases linearly in R. Similar revocation mechanisms relying on revocation lists of private keys or signatures have been proposed for direct anonymous attestation (DAA) settings [11,12]. Again, the computational effort as well as the signature size increases linearly with the revocation list. *Accumulator-based blacklist revocation* (BR-A) [3,13,24] applies (universal) cryptographic accumulators to allow for a compact and constant-size representation of RL as well as constant-size proofs to prove (non-)membership. *Blacklists of ordered credential-identifier pairs* [42] also lead to constant costs for signers and verifiers. But signers need to fetch an updated RL in the size of $\mathcal{O}(R)$ (and $\mathcal{O}(N)$ in predecessor schemes [46,47]) on each revocation, and the public key size is $\mathcal{O}(N)$ (or $\mathcal{O}(\sqrt{N})$ with significantly higher signing costs).

Verifier-Local Revocation. In *verifier-local revocation* (VLR) [8,44,45,62], verifiers when given a signature in question test if a certain relation holds for the signature and each entry on RL. The validity of the relation indicates that the signer has been revoked. Consequently, verifiers need to update RL on every revocation and a check during verification costs $\mathcal{O}(R)$ (typically group operations or even pairing evaluations). Although [8] proposes a constant-time revocation check, it only works if the same message and the same randomness is used for all signatures. This, however, is only reasonable for specific applications like DAA. Other disadvantages of VLR are that signatures of revoked members become linkable for all verifiers, *i.e.*, it lacks backward unlinkability, and that anyone in possession of RL can link signatures of revoked users. Although this can be fixed and backward unlinkability can be added, e.g., by introducing time intervals [44], this still adds additional non-trivial overhead. Besides, [22] proposed time-token dependent linking which can also be applied for VLR. However, signatures become publicly linkable (without any trapdoor information) if users sign more than once per time period, a separate RL must be maintained for each time period, and RLs need to be recomputed entirely for each time period since the revocation tokens of users change in each time period. While one could simply encrypt these revocation tokens to be decryptable by revocation authorities only, *i.e.*, to prevent public linkability, this would increase the signature size in part due to additional zero-knowledge proofs. In contrast, our approach preserves the privacy of signers and does not increase the signature size as it relies on the information already available in standard group signature schemes. Chow et al. [18] proposed a similar concept for membership revocation in ID-based ring signatures, a related but different concept. *Group signatures with probabilistic revocation* (GSPR) [38] allow for constant-time revocation checks at the expense of probabilistic revocation guarantees. However, in contrast to VLR the signer has to perform $\mathcal{O}(m)$ expensive operations, where m is a fixed value representing the number of signatures that can be issued by a signer before signatures become publicly linkable. Consequently, there is a trade-off between the storage/computational requirements for signers and the requirement for performing the group setup phase again. Moreover, the size of the group public key is $\mathcal{O}(m)$ and the GM needs to process $\mathcal{O}(m)$ user-specific tokens in order to update RL.

Revocation Mechanisms for Standard Model GS. For the sake of completeness we want to mention that there are also various revocation mechanisms designed to be compatible with the Groth-Sahai proof system [28] (instead of relying on Σ-protocols and the random oracle model). State-of-the-art mechanisms are due to Libert, Peters, and Yung (LPY) [40], which rely on the ciphertext of a broadcast encryption scheme as a RL. Later, LPY [39] has been improved to achieve constant size group signing keys. Attrapadung et al. (AEHS) [2] further reduced the revocation list to a constant size. However, signature sizes for LPY are about 100 and 144 group elements respectively, and AEHS produces even larger signatures. Thus, we exclude these revocation mechanisms from our comparison below, since we put a focus on group signature schemes that allow signers to be executed in resource-constrained environments as will be discussed in our motivating example later in this section.

Our Proposal (LBR). Existing revocation mechanisms are either inefficient or, in the worst case, even impossible to be implemented in resource-constrained environments, which is why revocation has been identified as the major bottleneck of state-of-the-art GSSs [41]. We address this problem by introducing *linking-based revocation* (LBR). LBR allows for a constant-time revocation mechanism that can be generically applied to existing GSSs, and in particular PB-GSSs [21, 29–32, 49] following the sign-and-encrypt-and-prove (SEP) paradigm [14, 36]. Essentially, we rely on the feature of controllable linkability [6, 29–31, 56] that allows a dedicated entity to determine whether two signatures have been produced by the same (anonymous) signer. By replacing the used public key encryption scheme of a GSS with its AoN-PKEET* variant (cf. Section 3.1), this feature can be added to GSSs following the SEP paradigm generically. The idea of LBR is that, in analogy to the online certificate status protocol (OCSP) [54] which is widely used for certificate revocation checks in the PKIX [19] setting[1], an online party can be contacted for revocation checks. To obtain robustness against compromise of online authorities, we introduce the feature of *distributed controllable linkability*, which may be of independent interest. When applying distributed controllable linkability to revocation, it allows RAs to anonymously link a given signature—with the cooperation of at least two linking authorities—against anonymous revocation tokens on RL. An additional optimization even allows for constant-time revocation checks.

In contrast to existing revocation mechanisms, our mechanism is transparent for signers and verifiers. Most importantly, LBR is efficient in the sense that (1) no key updates or additional computations are required for signers, (2) no expensive local revocation checks are required for verifiers, and (3) neither the signature size nor the key size increases. While all existing revocation approaches require signers and/or verifiers to fetch (possibly large) RLs from time to time, our mechanism relies on an always-online authority that is available for revocation checks. Although an *online* authority for such tasks might be considered as being *unconventional* or impractical at first, we believe that such an always-online requirement for specific authorities is absolutely reasonable.[2] In order to protect these RAs against attacks, we distribute the linking trapdoor required for the revocation checks to multiple entities. Consequently, an attacker would have to corrupt multiple entities to recover the linking trapdoor.

Comparison. Table 1 compares existing revocation mechanisms for practical group signature schemes in the random oracle model regarding their efficiency and practicality. For each mechanism, we compare the memory overhead for the group public key (GPK) and the signature as well as the computational overhead for updating keys/credentials, signature generation, and signature verification. Furthermore, we indicate the amount of information that needs to be fetched by

[1] A recent study [51] even states that OCSP is the most popular approach for revocation checks in the PKIX setting.

[2] An approach similar in spirit to our approach has recently also been discussed in the context of anonymous credential systems (cf. [60]).

signers and verifiers in case of revocation. As some schemes require both signers and verifiers to fetch updates, we also indicate whether these updates must be synchronized, *i.e.*, whether both parties need to have the same update-version as otherwise valid signatures cannot be computed and verified. Last but not least, we indicate whether signers and verifiers must be online for the revocation mechanism to work, where $\Diamond$ means semi-online, *i.e.*, signers and verifiers can decide when to go online to fetch the necessary updates.

Table 1. Comparison of Revocation Mechanisms.

Type	**Overhead memory**		**Overhead time**			**Updates**		**Synchronized**	**Online**	
	GPK	Signature	Update (signer)	Sign	Verify	Signers	Verifiers		Signers	Verifiers
CUR [7,31]	–	–	$\mathcal{O}(R)$	–	–	$\mathcal{O}(R)$	$\mathcal{O}(1)$	✓	$\Diamond$	$\Diamond$
BR-C [10]	–	$\mathcal{O}(R)$	–	$\mathcal{O}(R)$	$\mathcal{O}(R)$	$\mathcal{O}(R)$	$\mathcal{O}(R)$	✓	$\Diamond$	$\Diamond$
BR-ID [42]	$\mathcal{O}(\sqrt{N})$	$\mathcal{O}(1)$	–	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(R)$	$\mathcal{O}(1)$	✓	$\Diamond$	$\Diamond$
BR-A [3,13,24]	–	$\mathcal{O}(1)$	–	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	✓	$\Diamond$	$\Diamond$
VLR [8]	–	–	–	–	$\mathcal{O}(R)$	–	$\mathcal{O}(R)$	–	✘	$\Diamond$
GSPR [38]	$\mathcal{O}(m)$	$\mathcal{O}(1)$	–	$O(m)$	$\mathcal{O}(1)$	–	$\mathcal{O}(1)$	–	✘	$\Diamond$
LBR	–	–	–	–	$\mathcal{O}(1)$	–	–	–	✘	✓

Although VLR seems to provide similar advantages and features as LBR, our approach of LBR only allows RAs to link signatures, which is not the case within VLR as users can link signatures themselves. Thus, LBR overcomes the delicate issue of revoked members losing their anonymity. In addition, our proposed revocation mechanism can be generically applied to many PB-GSSs following the SEP paradigm, which covers a large class of state-of-the-art and practically efficient GSSs. As we will see below, our approach of LBR provides dedicated advantages and superior features for specific scenarios.

Motivating Example. As pairing-based cryptography has been optimized for resource-constrained devices [15,16,27,33,52,59], PB-GSSs have become entirely practical, at least when considering the performance of signature generation only. For instance, GSSs have been proposed as a privacy-preserving mechanism for public transport systems [34]. Their application allows passengers to anonymously prove possession of a valid ticket, but the service provider cannot identify passengers. Still, revocation of misbehaving passengers by invalidating tickets must be possible and these revocations should not affect other tickets in any way. Clearly, frequent updates through authenticated channels between the tickets and the service provider are impractical, as they would affect the valid tickets. Besides, performance is a crucial issue and, hence, the invalidation of one ticket should not lead to additional computations for the remaining (valid) tickets. While VLR might be a possible solution to overcome these problems, public transport systems usually support tickets with a limited validity, *i.e.*, 1-hour tickets, daily tickets, monthly tickets, and yearly tickets. Such tickets must be immediately revoked as soon as their validity ends and, thus, immediate revocation of tickets must be efficiently possible. Hence, VLR still faces the following problems. (1) RLs lead to $\mathcal{O}(R)$ computational effort for verifiers which is especially daunting in case of large RLs, and (2) RLs change frequently and must be distributed in a timely manner, *i.e.*, immediately after the revocation of one ticket, to many verifiers. Clearly, LBR overcomes these issues as it gets rid of the computational overhead for signers as well as verifiers and the need to communicate any revocation updates to signers and verifiers. Applying LBR allows the service provider to implement an existing PB-GSS (e.g. [21, 29–31, 49]) as a means to prove possession of a valid ticket. Turnstiles and gates that check the validity of a ticket are connected to the revocation authority. For each ticket to be verified, turnstiles request the revocation check via specifically deployed RAs and depending on the returned decision, access is either granted or denied. Considering some of the biggest metro systems around the world with several hundred millions of served passengers per year, e.g., Beijing, Moscow, and NY City, the efficiency of the used revocation mechanism is of utmost importance. Besides, also the European Union demands for privacy protection of individuals and the principle of data minimization in transportation systems within the EU Directive 2010/40. Hence, GSSs will likely play an important role in the future and efficient revocation mechanisms will be required.

3 Preliminaries

Let $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle \hat{g_2} \rangle$, and $\mathbb{G}_T$ be cyclic groups of prime order p. We write elements in $\mathbb{G}_2$ as $\hat{g}, \hat{v}$, etc. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a map, such that $e(u^a, \hat{v}^b) = e(u, \hat{v})^{ab}$ for all $u \in \mathbb{G}_1$, $\hat{v} \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$. Additionally, we require $e(g_1, \hat{g_2}) \neq 1$ and e to be efficiently computable. If $\mathbb{G}_1 = \mathbb{G}_2$, then e is called *symmetric* (Type 1) and *asymmetric* (Type 2 or Type 3) otherwise. For Type 2 pairings there is an efficiently computable isomorphism $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$, whereas for Type 3 pairings no such efficient isomorphism is known. We (informally) state the used assumptions below.

DDH. Let $\mathbb{G}$ be a cyclic group of prime order p and g a generator. The DDH assumption states that given (g, g^a, g^b, g^c) it is hard to decide whether $ab = c$.

(S)XDH. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three cyclic groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ a pairing. The (S)XDH assumption states that the DDH assumption holds in $\mathbb{G}_1$ (and $\mathbb{G}_2$).

A function $\epsilon : \mathbb{N} \rightarrow \mathbb{R}^+$ is called negligible if for all $c > 0$ there is a k_0 such that $\epsilon(k) < 1/k^c$ for all $k > k_0$. We use ϵ to denote such a negligible function.

3.1 All-or-Nothing Public Key Encryption with Equality Tests

All-or-nothing public key encryption with equality tests (AoN-PKEET) [57] allows entities in possession of a trapdoor to perform equality tests on ciphertexts without learning the underlying plaintexts. A modification denoted as AoN-PKEET* [56] allows IND-CPA security against outsiders and requires compatibility with efficient zero-knowledge proofs of knowledge (ZKPoK) of plaintexts. Such an AoN-PKEET* scheme (KeyGen, Enc, Dec, Aut, Com) is a conventional (at least IND-CPA secure) public key encryption scheme (KeyGen, Enc, Dec) (compatible with efficient ZKPoK) augmented by two algorithms Aut and Com.

$\mathsf{KeyGen}(1^\lambda)$: The key generation algorithm takes a security parameter λ, and generates a public-private key pair $(\mathsf{pk}, \mathsf{sk})$ used for encryption and decryption operations.

$\mathsf{Enc}(\mathsf{pk}, m)$: The encryption algorithm takes the public key pk, and a message m, and returns the encryption c of m under pk.

$\mathsf{Dec}(\mathsf{sk}, c)$: The decryption algorithm takes the private key sk, and a ciphertext c, and returns the message m.

$\mathsf{Aut}(\mathsf{sk})$: Takes the private decryption key sk of the public key encryption scheme and returns the trapdoor tk used for equality tests.

$\mathsf{Com}(T, T', \mathsf{tk})$: Takes two ciphertexts (T, T') and a trapdoor tk and returns `true` if T and T' encrypt the same (unknown) message and `false` otherwise.

Definition 1 **([56])** *An AoN-PKEET* scheme is secure if it is sound, provides* OW-CPA *security against Type-I adversaries (trapdoor holders) and if the underlying encryption scheme provides* IND-CPA/IND-CCA2 *security against Type-II adversaries (outsiders).*

Soundness requires correctness of the public key encryption scheme and for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ one requires that $\mathsf{Com}(\mathsf{Enc}(\mathsf{pk}, m), \mathsf{Enc}(\mathsf{pk}, m'), \mathsf{Aut}(\mathsf{sk})) = \texttt{true}$ if and only if $m = m'$. Subsequently, we provide definitions for OW-CPA as well as IND-CPA/IND-CCA2 security, where $\mathcal{M}$ denotes the message space of a scheme.

Definition 2 **(OW-CPA security)** *An* AoN-PKEET* *scheme is* OW-CPA *secure against Type-I adversaries, if for all PPT adversaries $\mathcal{A}$ and security parameters λ there is a negligible function ϵ such that:*

$$\Pr\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{tk} \leftarrow \mathsf{Aut}(\mathsf{sk}), m \xleftarrow{R} \mathcal{M}, \\ c \leftarrow \mathsf{Enc}(\mathsf{pk}, m), m^* \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{tk}, c)\end{array} \ : \ m^* = m\right] \leq \epsilon(\lambda).$$

Definition 3 **(IND-CPA/IND-CCA2 security)** *An* AoN-PKEET* *scheme is* IND-CPA/IND-CCA2 *secure against Type-II adversaries, if for all PPT adversaries $\mathcal{A}$ and security parameters λ there is a negligible function ϵ such that:*

$$\Pr\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{tk} \leftarrow \mathsf{Aut}(\mathsf{sk}), \\ (m_0, m_1, s) \leftarrow \mathcal{A}^{\mathcal{O}_1}(\mathsf{pk}), b \xleftarrow{R} \{0,1\}, c \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b) \\ b^* \leftarrow \mathcal{A}^{\mathcal{O}_2}(\mathsf{pk}, c, s)\end{array} \ : \ b^* = b\right] \leq 1/2 + \epsilon(\lambda)$$

where $m_0, m_1 \in \mathcal{M}$, and

$$\begin{array}{lll}\mathcal{O}_1(\cdot) = \perp & \textit{and } \mathcal{O}_2(\cdot) = \perp & \textit{for } \mathsf{IND\text{-}CPA} \\ \mathcal{O}_1(\cdot) = \mathcal{O}_{\mathsf{Dec}} & \textit{and } \mathcal{O}_2(\cdot) = \mathcal{O}_{\mathsf{Dec}} & \textit{for } \mathsf{IND\text{-}CCA2}\end{array}$$

and $\mathcal{O}_{\mathsf{Dec}}$ represents the decryption oracle, and $\mathcal{A}$ is not allowed to query the decryption oracle for the challenge ciphertext c.

Instantiation based on ElGamal Encryption. For reasons of simplicity we will demonstrate our approach based on ElGamal encryption (instead of its twin variants [25,48,53] as demonstrated in Section 6), but we stress that it also works for other encryption schemes used in the pairing setting like linear ElGamal [7] (and its corresponding twin variant) or Cramer-Shoup encryption [20]. Nevertheless, if AoN-PKEET* is instantiated with ElGamal encryption and assuming the private decryption key to be $\xi \in \mathbb{Z}_p$ and the corresponding public key $h = g^{\xi} \in \mathbb{G}_1$, the resulting ciphertext is $(T_1 = g^{\alpha}, T_2 = m \cdot h^{\alpha}) \in \mathbb{G}_1^2$ for a random $\alpha \in \mathbb{Z}_p$. Given two ciphertexts T and T', and the trapdoor key $\mathsf{tk} = (\hat{r}, \hat{s} = \hat{r}^{\xi}) \leftarrow \mathsf{Aut}(\xi)$ for a random $\hat{r} \in \mathbb{G}_2$, the equality test on the encrypted messages can be performed via the $\mathsf{Com}(T, T', \mathsf{tk})$ algorithm of the AoN-PKEET* scheme by evaluating whether the following holds: $e(T_2, \hat{r}) \cdot e(T_1, \hat{s})^{-1} = e(T_2', \hat{r}) \cdot e(T_1', \hat{s})^{-1}$.

3.2 Sign-and-Encrypt-and-Prove Paradigm

GSSs following the SEP paradigm consist of the following three building blocks: (1) a secure signature scheme $\mathcal{DS} = (\mathsf{KeyGen_s}, \mathsf{Sign}, \mathsf{Verify})$; (2) an at least IND-CPA secure public key encryption scheme $\mathcal{AE} = (\mathsf{KeyGen_e}, \mathsf{Enc}, \mathsf{Dec})$; and (3) non-interactive zero-knowledge proofs of knowledge (NIZKPKs). For schemes in the ROM latter are honest-verifier zero-knowledge proofs of knowledge made non-interactive using the Fiat-Shamir transform (denoted as signatures of knowledge (SoK) subsequently).

The group public key gpk consists of the public encryption key pk_e, and the signature verification key pk_s. The master opening key mok is the decryption key sk_e and the master issuing key mik is the signing key sk_s. During the joining procedure a user i sends $f(x_i)$ to the issuer, where $f(\cdot)$ is a one-way function applied to a secret x_i. The issuer returns a signature $\mathsf{cert} \leftarrow \mathsf{Sign}(\mathsf{sk}_s, f(x_i))$ as the user's certificate. A group signature $\sigma = (T, \pi)$ for a message M consists of a ciphertext $T \leftarrow \mathsf{Enc}(\mathsf{pk}_e, \mathsf{cert})$ and the following SoK π:

$$\pi \leftarrow \mathsf{SoK}\{(x_i, \mathsf{cert}) : \mathsf{cert} = \mathsf{Sign}(\mathsf{sk}_s, f(x_i)) \ \wedge \ T = \mathsf{Enc}(\mathsf{pk}_e, \mathsf{cert})\}(M).$$

The above description provides an intuition for the SEP approach and there exist different variations, e.g., sometimes cert is computed for x_i instead of $f(x_i)$ (which, however, does not yield constructions providing non-frameability), or T may represent an encryption of $f(x_i)$ or $g(x_i)$ for some one-way function $g(\cdot)$. This, however, is not important in the context of controllable linkability, where it is only required that T contains the encryption of a constant, per-user unique value cert.

3.3 Threshold Secret Sharing

A (t, n)-threshold secret sharing scheme allows to distribute a secret s to n parties in a way such that it requires the cooperation of at least t parties to recover s, while any set of up to $t-1$ parties learns nothing about s. An elegant and famous (t, n)-threshold scheme based on polynomial interpolation has been proposed by Shamir [55]. Here, to share a secret s, one chooses a random polynomial $F(x) = s + \sum_{\ell=1}^{t-1} a_\ell \cdot x^\ell$ of degree $t-1$ such that $F(0) = s$ over some finite field. Shares are of the form $(i, F(i))$ and any subset of size at most $t-1$ will learn no information about s. Given shares $(i, F(i)) \in \mathcal{I}$ such that $|\mathcal{I}| \geq t$, s can however be efficiently recovered via $s = F(0) = \sum_{i\in\mathcal{I}} c_i \cdot F(i)$, where $c_i = \prod_{j\in\mathcal{I}, j\neq i} \frac{j}{j-i}$ are the corresponding Lagrange coefficients. We will use this scheme to share a group element from a prime order group (cf. [4]).

3.4 Group Signatures with Controllable Linkability

Hwang et al. [29–31] introduced a model for GSSs with controllable linkability that builds upon the well-established BSZ model [5] (although, Hwang et al. use a weaker notion of anonymity). The group manager is logically split into (1) an opening authority capable of opening signatures, (2) an issuing authority capable of issuing signing keys to group members, and (3) a linking authority capable of linking signatures, *i.e.*, an authority that can determine whether or not two signatures have been issued by the *same anonymous* signer. However, the linking key does not allow to actually identify the signer, which means that the linking authority is strictly less powerful than the opening authority. We denote the keys of these authorities as master opening key (mok), master issuing key (mik), and master linking key (mlk), respectively. A GSS with controllable linkability is a tuple $\mathcal{GS}\text{-}\mathcal{CL} = (\mathsf{GkGen}, \mathsf{UkGen}, \mathsf{Join}, \mathsf{Issue}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge}, \mathsf{Link})$ of PPT algorithms as defined in [5, 29–31] and recalled subsequently.

$\mathsf{GkGen}(1^\lambda)$**:** On input a security parameter λ, this algorithm generates the public parameters and outputs a tuple $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk})$, representing the group public key, the master opening key, the master issuing key, and the master linking key.

$\mathsf{UkGen}(1^\lambda)$**:** On input a security parameter λ, this algorithm generates a user key pair $(\mathsf{usk}_i, \mathsf{upk}_i)$.

$\mathsf{Join}(\mathsf{usk}_i, \mathsf{upk}_i)$**:** On input the user's key pair $(\mathsf{usk}_i, \mathsf{upk}_i)$, this algorithm interacts with Issue and outputs the group signing key gsk_i of user i.

$\mathsf{Issue}(\mathsf{gpk}, \mathsf{mik}, \mathbf{reg})$**:** On input the group public key gpk, the master issuing key mik, and the registration table $\mathbf{reg}$, this algorithm interacts with Join to add user i to the group.

$\mathsf{GSig}(\mathsf{gpk}, M, \mathsf{gsk}_i)$**:** On input the group public key gpk, a message M, and a user's secret key gsk_i, this algorithm outputs a group signature σ.

$\mathsf{GVf}(\mathsf{gpk}, M, \sigma)$**:** On input the group public key gpk, a message M, and a signature σ, this algorithm verifies whether the signature σ is valid with respect to the message M and the group public key gpk and outputs `true` if the verification succeeds and `false` otherwise.

$\mathsf{Open}(\mathsf{gpk}, \mathbf{reg}, M, \sigma, \mathsf{mok})$**:** On input the group public key gpk, the registration table $\mathbf{reg}$, a message M and a valid signature σ corresponding to this message, and the master opening key mok, this algorithm returns the signer i together with a publicly verifiable proof τ attesting the validity of the claim and $\perp$ otherwise.

$\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i, \mathsf{upk}_i, \tau)$**:** On input the group public key gpk, a message M, a valid signature σ, the claimed signer i, the user's public key upk_i as well as a proof τ, this algorithm returns `true` if τ is a valid proof that i produced σ and `false` otherwise.

$\mathsf{Link}(\mathsf{gpk}, M, \sigma, M', \sigma', \mathsf{mlk})$**:** On input the group public key gpk, a message M and valid signature σ as well as a message M' and valid signature σ', and the master linking key mlk, this algorithm returns `true` if both signatures stem from the same signer and `false` otherwise.

Security Properties for GSs with Controllable Linkability. For a GSS with controllable linkability to be secure it needs to satisfy the following properties (cf. [5, 29, 31] for a formal description).

- **Anonymity**: The identity of the signer can only be determined by the authority in possession of the master opening key.
- **Traceability**: The opening authority must be able to open a valid signature and to prove the corresponding claim.
- **Non-frameability**: An adversary should not be able to prove that an honest user generated a signature unless this user indeed produced this signature.
- **Linkability**: The master linking key should neither be useful to gain any information for opening a signature nor for generating opening proofs. Furthermore, colluding parties—including users, the linker, and/or the opener—should not be able to generate pairs of messages and signatures with contradicting open and link decisions.

3.5 Concepts Related to Controllable Linkability

The following two concepts are related to controllable linkability and thus we discuss their suitability to implement linking-based revocation (LBR) subsequently. The first concept does not qualify for LBR due to privacy concerns (public linkability of signatures). The second concept qualifies for LBR, but the underlying linking mechanisms always requires computations linear in the number of revoked members. We briefly discuss both approaches below. For a more detailed comparison of these concepts we refer the interested reader to [56].

Linkable Group Signatures. Constructions of GSSs relying on tracing-by-linking [58, 61] do not employ the SEP paradigm and, hence, no authority can open a given signature. Only if a member signs more than k times, signatures of this member become publicly traceable. Similarly, link-but-not-trace GSSs [43] allow to publicly link signatures, while opening requires all users to prove that a signature has not been produced by them (disavowing), which is clearly not possible for the member who actually produced this message. Both of these approaches allow to publicly link signatures and are thus not appropriate candidates for a revocation mechanism we are envisioning.

Traceable Signatures. Traceable signatures [35] are a variant of group signatures that allow the group manager to publish a tracing trapdoor for any group member that can be used to trace signatures of the respective member. Consequently, they could be used in a similar manner for revocation as controllable linkability. Namely, on revocation one could call the Reveal algorithm for the revoked user and provide the respective tracing trapdoor to the RA. Given a signature, the RA uses all the tracing trapdoors (representing the revocation list) and runs the Trace algorithm on the given signature and every tracing trapdoor from the list. However, while we achieve a constant-time revocation check, traceable signatures would always require a linear check.

4 Building Blocks for GSs with Linking-Based Revocation

Subsequently, we briefly outline the high-level idea of our proposed revocation mechanism. Afterwards, we introduce the necessary building blocks and modifications, *i.e.*, we show how to achieve constant-time revocation checks and we also introduce the feature of distributed controllable linkability.

4.1 High-Level Idea of GSs with Linking-Based Revocation

We recall that the generic compiler in [56] allows to add controllable linkability to PB-GSSs following the SEP paradigm that are secure in the BSZ [5] model. Essentially, this generic compiler replaces the used encryption scheme (used to encrypt membership certificates) with its AoN-PKEET* variant, which allows to determine whether two ciphertexts encrypt the same plaintext, without learning the plaintexts. Thereby, controllable linkability allows a dedicated authority to determine whether two signatures have been issued by the same *unknown* signer by performing an equality test on the encrypted membership certificates. We stress that the dedicated approaches to construct group signatures with controllable linkability in [29–31] implicitly use the same idea and thus can also be used in combination with our revocation approach. Technically, they do not directly apply AoN-PKEET* to the membership certificate, but another user-related value. Nevertheless, one can apply our subsequently discussed ideas analogously.

Based on the concept of controllable linkability, the idea of linking-based revocation is as follows. A verifier first verifies a given signature before contacting a dedicated RA for the revocation check. Note that this also means that the signature verification is decoupled from the actual revocation check and, thus, revocation checks can also be postponed as in case of OCSP requests in the PKIX setting. The RA is given the master linking key, a revocation list, e.g., a list of signatures of revoked members, and a signature in question. For the revocation check, the RA links the given signature against all entries on RL. If any of these signatures links, the corresponding signer has been revoked. Figure 1 illustrates this basic approach, which is, however, rather naive for the following reasons.

1. The revocation check is linear in the size of RL, *i.e.*, has cost $\mathcal{O}(R)$.

2. If an attacker compromises the RA and steals the linking key, she would be able to link any two signatures, which is clearly not desired.

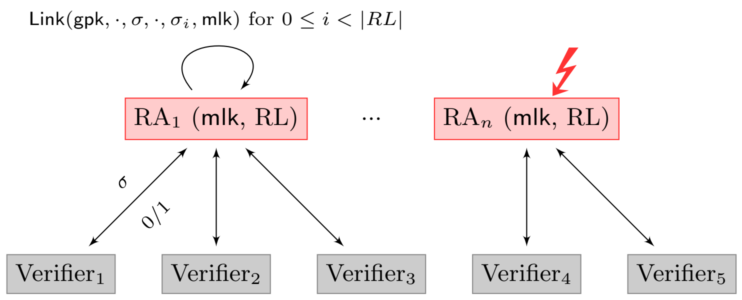

Fig. 1. Naive (insecure) instantiation of linking-based revocation: An attacker can steal the linking key mlk by compromising a single revocation authority.

Subsequently, we deal with these issues and gradually introduce the necessary modifications to achieve (1) constant-time revocation checks, and (2) to remove the single point of attack by distributing the linking key among multiple entities.

4.2 Constant-Time Revocation Checks

To obtain constant-time revocation checks, we modify the $\mathsf{Com}(T, T', \mathsf{tk})$ algorithm of the AoN-PKEET* scheme in a way that it does no longer decide whether two ciphertexts encrypt the same message, but instead returns a value—the *revocation token*—that is computed from a given ciphertext T and the trapdoor tk. We stress that we cannot generically decide for any AoN-PKEET* whether this is possible, but it is in fact possible for all natural ElGamal-style AoN-PKEET* schemes in the pairing setting. For instance, for conventional ElGamal encryption this yields revocation tokens of the form $e(T_2, \hat{r}) \cdot e(T_1, \hat{s})^{-1} = e(m, \hat{r})$. Subsequently we denote such an invocation as $\mathsf{t} \leftarrow \mathsf{Com}(T, \perp, \mathsf{tk})$.

Security Definition. In order to reason about the security of such a mechanism when applied to revocation, we introduce the notion of token indistinguishability. Token indistinguishability considers an adversary that does not know the trapdoor tk but for a ciphertext $T = (T_1, T_2)$ on any message m observes tokens t of the form $e(T_2, \hat{r}) \cdot e(T_1, \hat{t})^{-1} = e(m, \hat{r})$. This information, however, does not allow the adversary to reason about any tokens seen in the future.

Definition 4 (Token Indistinguishability) *An* AoN-PKEET* *scheme is* $\mathsf{T\text{-}IND}$*, if for all PPT adversaries $\mathcal{A}$ and security parameters λ there is a negligible* *function ϵ such that:*

$$\Pr\left[\begin{array}{l}(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{tk} \leftarrow \mathsf{Aut}(\mathsf{sk}),\\ s \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{RMsg}}}(\mathsf{pk}), m \xleftarrow{R} \mathcal{M}, c \leftarrow \mathsf{Enc}(\mathsf{pk}, m),\\ b \xleftarrow{R} \{0,1\}, \mathsf{t}_0 \leftarrow \mathsf{Com}(c, \perp, \mathsf{tk}), \mathsf{t}_1 \xleftarrow{R} \mathcal{T}\\ b^* \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{RMsg}}}(\mathsf{pk}, m, \mathsf{t}_b, s)\end{array} \; : \; b^* = b\right] \leq 1/2 + \epsilon(\lambda)$$

where $\mathcal{M}$ represents the message space, $\mathcal{T}$ represents the token space, and $\mathcal{O}_{\mathsf{RMsg}}$ represents the oracle to generate random messages and corresponding tokens (m_i, t_i), such that $m_i \xleftarrow{R} \mathcal{M}$ and $\mathsf{t}_i \leftarrow \mathsf{Com}(\mathsf{Enc}(\mathsf{pk}, m_i), \perp, \mathsf{tk})$.

Lemma 1. *Under the DDH assumption, AoN-PKEET* based on ElGamal in $\mathbb{G}_1$ in an XDH setting is* T-IND.

Proof (Lemma 1). Given an adversary $\mathcal{A}$ that breaks the T-IND of AoN-PKEET*, we show how to construct an adversary $\mathcal{B}$ against DDH. Let (g, g^a, g^b, g^c) be a DDH instance given to $\mathcal{B}$. $\mathcal{B}$ randomly generates a private key sk and a corresponding public key pk, and sets $\mathsf{tk} \leftarrow \mathsf{Aut}(\mathsf{sk})$, *i.e.*, $\mathcal{B}$ implicitly sets $\hat{r} = \hat{g}^b$. $\mathcal{A}$ is now allowed to query $\mathcal{O}_{\mathsf{RMsg}}$, which $\mathcal{B}$ answers as $(g^{m_i}, e((g^b)^{m_i}, \hat{g}))$ for a random $m_i \in \mathbb{Z}_p$. Eventually, $\mathcal{A}$ receives the challenge $(g^a, e(g^c, \hat{g}))$ and outputs its guess. It is clear that if the DDH instance is valid, then the challenge represents a valid message-token tuple and is an independent and random element otherwise. Thus, we perfectly simulate the T-IND game for $\mathcal{A}$ and it is clear that whenever $\mathcal{A}$ breaks T-IND we can break DDH with the same probability. □

4.3 Distributed Controllable Linkability

Due to the fact that our proposed revocation mechanism relies on an always-online revocation authority, the attack surface is significantly larger than in case of an offline authority. Essentially, we want to ensure that an attacker cannot steal the master linking key mlk by compromising such an authority. In order to prevent such a single point of failure, we thus introduce threshold AoN-PKEET* which then enables us to realize distributed controllable linkability. Thereby, we reduce the trust in the linking authority and also obtain more robustness. In a similar manner Ghadafi [26] introduced distributed tracing, such that multiple opening authorities must cooperate in order to open a signature.

The basic idea is to distribute the trapdoor $\mathsf{tk} \leftarrow \mathsf{Aut}(\mathsf{sk})$ of an AoN-PKEET* primitive among n entities using a (t, n)-secret sharing scheme. Then, the cooperation of at least t authorities is required to recover the trapdoor tk or to employ the trapdoor to perform equality tests on encrypted data.

Formal Model. We define threshold AoN-PKEET* as a tuple of algorithms $\mathcal{T}\text{-}\mathcal{PKEQ}^* = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Aut}, \mathsf{DKAut}, \mathsf{TShare}, \mathsf{TSCom})$, where DKAut is an algorithm that computes the shares for the trapdoor key, TShare is an algorithm to compute the corresponding trapdoor shares for given ciphertexts, and TSCom is an algorithm to perform the plaintext equality test based on a given set of shares.

$\mathsf{DKAut}(\mathsf{tk}, t, n)$**:** Takes a trapdoor key tk, a threshold t, and a number of total shares n, and returns trapdoor shares $(\mathsf{tk}_i)_{i=1}^n$, such that a subset of at least t entities is required to perform equality tests.

$\mathsf{TShare}(T, T', \mathsf{tk}_i)$**:** Takes two ciphertexts (T, T') and a trapdoor share tk_i, and returns corresponding shares C_i and C'_i for the equality test.

$\mathsf{TSCom}(\{C_i, C'_i\}_{i\in\mathcal{I}})$**:** Given a set of shares $\{C_i, C'_i\}_{i\in\mathcal{I}}$ with $|\mathcal{I}| \geq t$, the algorithm combines the shares to perform the plaintext equality test and returns `true` if T and T' encrypt the same (unknown) message and `false` otherwise.

Similarly to how the $\mathsf{Com}(T, \perp, \mathsf{tk})$ algorithm has been adapted to return an anonymous revocation token t for a given ciphertext T, the TShare and TSCom algorithms can be adapted to return appropriate shares and the corresponding revocation token, respectively. We denote an invocation that returns the corresponding shares as $\mathsf{TShare}(T, \perp, \mathsf{tk}_i)$ and an invocation that combines these shares to compute the revocation token as $\mathsf{TSCom}(\{C_i, \perp\})$.

Instantiation Based on ElGamal and Shamir's Secret Sharing. Again, we assume a conventional ElGamal-based AoN-PKEET* in a bilinear map setting, where the private key is $\mathsf{sk} = \xi \in \mathbb{Z}_p$ and the trapdoor for plaintext equality tests is $\mathsf{tk} = (\hat{r}, \hat{s} = \hat{r}^\xi) \in \mathbb{G}_2^2$. We omit the KeyGen, Enc, Dec algorithms for the sake of brevity and only present the relevant algorithms below.

$\mathsf{DKAut}(\mathsf{tk}, t, n)$**:** Given a trapdoor key $\mathsf{tk} = (\hat{r}, \hat{s} = \hat{r}^\xi)$, a threshold t, and a total number of shares n, it computes the shares $(\mathsf{tk}_i)_{i=1}^n$. Therefore, it computes a polynomial $F(x) = \hat{r} \cdot \prod_{\ell=1}^{t-1} \hat{f}_\ell^{x^\ell}$ for random $\hat{f}_\ell \in \mathbb{G}_2$, e.g., $\hat{f}_\ell = \hat{g}^r$ for random $r \in \mathbb{Z}_p$. Similarly, it computes a polynomial $G(x) = \hat{s} \cdot \prod_{\ell=1}^{t-1} \hat{g}_\ell^{x^\ell}$ for random $\hat{g}_\ell \in \mathbb{G}_2$. Finally, it returns the shares $(\mathsf{tk}_i = (i, \hat{r}_i \leftarrow F(i), \hat{s}_i \leftarrow G(i)))_{i=1}^n$.

$\mathsf{TShare}(T, T', \mathsf{tk}_i)$**:** Given two ciphertexts $(T, T') = ((T_1, T_2), (T'_1, T'_2))$ as well as a share of the trapdoor tk_i, it computes and returns the comparison shares $C_i = e(T_2, \hat{r}_i)$ and $D_i = e(T_1, \hat{s}_i)$ and $C'_i = e(T'_2, \hat{r}_i)$ and $D'_i = e(T'_1, \hat{s}_i)$.

$\mathsf{TSCom}(\{C_i, D_i, C'_i, D'_i\}_{i\in\mathcal{I}})$**:** Given a set of comparison shares $\{C_i, D_i, C'_i, D'_i\}_{i\in\mathcal{I}}$ with $|\mathcal{I}| \geq t$, the algorithm combines the shares to perform the plaintext equality test. Therefore, it computes S and S' as follows:

$$S = \prod_{i\in\mathcal{I}} C_i^{L_i} \cdot \left(\prod_{i\in\mathcal{I}} D_i^{L_i}\right)^{-1} \qquad S' = \prod_{i\in\mathcal{I}} C_i'^{L_i} \cdot \left(\prod_{i\in\mathcal{I}} D_i'^{L_i}\right)^{-1}$$

where $L_i = \prod_{j\in\mathcal{I}} \frac{j}{j-i}$ for $j \neq i$ are the Lagrange coefficients. Finally, it returns `true` if $S = S'$ and `false` otherwise.

The correctness of the above construction can be seen by inspection. Furthermore, the notion of token indistinguishability ($\mathsf{T\text{-}IND}$) as defined in Section 4.2 also holds for the threshold variant of AoN-PKEET*.

5 GSs with Linking-Based Revocation

We now specify a GSS with linking-based revocation as a tuple $\mathcal{GS}\text{-}\mathcal{LBR}$ = (GkGen, UkGen, Join, Issue, GSig, GVf, Open, Judge, $\mathsf{CheckStatus}$, Revoke). Next, we outline the algorithms that change due to our modifications as well as the additional algorithms $\mathsf{CheckStatus}$ and Revoke.

$\mathsf{GkGen}(1^\lambda, t, n)$**:** On input a security parameter λ, a threshold t, and a total number of shares n, the algorithm outputs a tuple $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}, (\mathsf{mlk}_i)_{i=1}^n)$. First, it runs $(\mathsf{pk}_e, \mathsf{sk}_e) \leftarrow \mathsf{KeyGen}(1^\lambda)$ of the (t, n)-threshold AoN-PKEET* scheme, sets $\mathsf{mok} = \mathsf{sk}_e$, and integrates pk_e into gpk. Then it runs $(\mathsf{tk}_{\mathsf{pub}}, \mathsf{tk}_{\mathsf{priv}}) \leftarrow \mathsf{Aut}(\mathsf{mok})$, sets the master linking key $\mathsf{mlk} = (\mathsf{tk}_{\mathsf{pub}}, \mathsf{tk}_{\mathsf{priv}})$, and integrates $\mathsf{tk}_{\mathsf{pub}}$ into mik. Furthermore, it generates the shares $\mathsf{mlk}_i \leftarrow \mathsf{DKAut}(\mathsf{mlk}, t, n)$ for the n distributed linking authorities. The rest remains unchanged.

$\mathsf{GVf}(\mathsf{gpk}, M, \sigma)$**:** On input the group public key gpk, a message M, and a signature σ, the algorithm determines whether the signature σ is valid with respect to the message M and the group public key gpk. It returns `true` if the signature is valid and `false` otherwise.

$\mathsf{CheckStatus}(\mathsf{RL}, \mathcal{L}, \sigma)$**:** On input a revocation list RL containing anonymous revocation tokens, a set of existing linking authorities $\mathcal{L}$, and a signature σ, this algorithm determines the revocation status of the signer corresponding to signature σ. In order to determine the revocation status, it interacts with t linking authorities $\mathcal{L}_i \in \mathcal{L}$ via the TShare algorithm and retrieves the corresponding shares for the computation of the revocation token. Afterwards, it uses the TSCom algorithm to combine these shares and to retrieve the final revocation token. If the revocation token exists on the RL, the signer has been revoked and it returns `true`. Otherwise, it has not been revoked and it returns `false`.

$\mathsf{Revoke}(\mathsf{gpk}, \mathsf{mik}, \mathbf{reg}, \mathsf{RL}, i)$**:** On input of the group public key gpk, the master issuing key mik, the registration table $\mathbf{reg}$, the current revocation list RL, and a user i to be revoked[3], the algorithm computes the anonymous revocation token t_i corresponding to user i and adds it to the revocation list. It returns the updated revocation list $\mathsf{RL} = \mathsf{RL} \cup \{\mathsf{t_i}\}$.

5.1 Discussion and Security

Computation of Revocation Tokens. Staying with our conventional ElGamal example and considering revocation tokens which are of the form $e(T_2, \hat{r}) \cdot e(T_1, \hat{s})^{-1} = e(m, \hat{r})$, we observe that these tokens can be computed in two different ways.

Given m **and** $\hat{r}$**:** Given a message m, e.g., a user's certificate, and $\hat{r}$ allows to compute revocation tokens $\mathsf{t} = e(m, \hat{r})$. Thus, if the issuer is given access to

[3] Note that revocation can also be done based on a user's signature by means of mlk in which case the user's identity will not be required.

$\hat{r}$, the revocation token t can be computed with the information available during the Issue algorithm and added to the registration table **reg**.

Given $\sigma = (T, \pi)$ and $\mathsf{mlk} = (\hat{r}, \hat{r}^{\xi})$: Given a signature $\sigma = (T, \pi)$ and the master linking key mlk, such a revocation token t can also be computed on the fly, *i.e.*, $\mathsf{t} \leftarrow \mathsf{Com}(T, \perp, \mathsf{mlk})$. Thus, such a token can be computed directly from a given signature which allows for anonymous revocation of users as signatures need not be opened before revocation.

Note that if the revocation tokens are precomputed and stored in the registration table **reg**, then an attacker who manages to get in possession of the registration table **reg** and RL (but not necessarily the mlk) can conceptually identify (open) all signers on RL as the same tokens can be found in **reg**. This is not possible in case the revocation tokens are computed on the fly as in this case the attacker cannot link entries on RL to entries in the registration table **reg** since the revocation tokens do not yield any useful information (cf. $\mathsf{T\text{-}IND}$).

Revocation and Revocation Check. Eventually, in case of a revocation, the token—that can either be computed (1) by opening a signature first or (2) from the signature directly—is added to RL. The actual revocation check for a signature $\sigma = (T, \pi)$ then requires the computation of the token $\mathsf{t} \leftarrow \mathsf{Com}(T, \perp, \mathsf{mlk})$ and a simple look-up operation, *i.e.*, checking whether or not $\mathsf{t} \in \mathsf{RL}$, and consequently can be performed in time $\mathcal{O}(1)$.[4] Also note that, except for the party in possession of mlk, the anonymous revocation token does not yield any useful information and also does not endanger the privacy of signers.

Revocation Check with Threshold AoN-PKEET*. We point out that a RA in our setting does not hold the linking key mlk but always needs to contact a set of at least t linking authorities (over authenticated and confidential channels) to compute the required tokens. Thereby, we assume that no t linking authorities can be compromised. Consequently, in contrast to the naive approach, breaking into the (always-online) RA does not reveal the linking key. The only information that an attacker gains by compromising RAs is a list of revocation tokens t_i (and possibly the corresponding messages m_i and ciphertexts c_i). However, as already argued in Section 4.2, this does not allow the attacker to compromise the overall anonymity of the scheme as this information does not allow her to distinguish other tokens t from random. Although we only cover passive attacks, this is a reasonable model because in case a RA gets compromised, the corresponding authentication key will be revoked and replaced and the adversary can only behave passive.

Figure 2 illustrates the basic idea of our secure instantiation of linking-based revocation. A verifier first verifiers a given signature $\sigma = (T, \pi)$ via the GVf algorithm and afterwards wants to learn about the revocation status of this signature. Therefore, it interacts with a RA by means of $\mathsf{CheckStatus}$. The RA interacts with t LAs in order to retrieve the corresponding shares

[4] Hash tables allow to check whether or not $\mathsf{t} \in \mathsf{RL}$ in constant time. For instance, employing cuckoo hashing [50] allows for a worst-case complexity of $\mathcal{O}(1)$.

by means of $\{C_i, \perp\} \leftarrow \mathsf{TShare}(T, \perp, \mathsf{mlk}_i)$ and, afterwards, RA employs the $\mathfrak{t} \leftarrow \mathsf{TSCom}(\{C_i, \perp\}_{i \in \mathcal{I}})$ algorithm to combine these shares, which yields the revocation token $\mathfrak{t}$. After checking whether or not $\mathfrak{t}$ exists on RL, the RA returns the corresponding decision via CheckStatus.

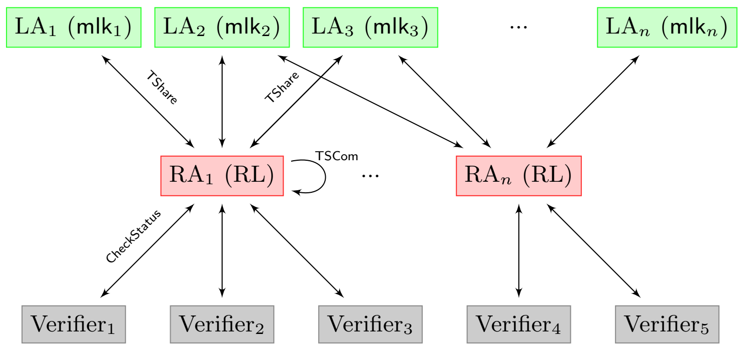

Fig. 2. Schematic of our secure instantiation of linking-based revocation.

As the RL as well as the learned tokens $\mathfrak{t}$ do not endanger the privacy of group members (cf. T-IND), the role of RAs can be distributed over multiple cloud services. Besides, we assume that no t LAs can be compromised at once and since the trapdoor shares mlk_i do not endanger the privacy of group members, these trapdoor shares mlk_i can also be safely distributed over multiple cloud services. Similar to traditional OCSP responses, the response from RAs must be signed. Although one could also employ the feature of verifiable controllable linkability [6] in order to prove that a given signature has or has not been revoked, this would add an additional overhead for the verifier, especially in case the signer has not been revoked. More specifically, if the signer is already revoked, the RA needs to perform a proof that the given signature can be linked to one specific entry on RL. In contrast, if the signer has not been revoked, the RA needs to prove that the given signature has not been produced by any entity on RL and, hence, a naive instantiation of verifiable controllable linkability means that the proof increases linearly with the size of RL. In addition, such a naive instantiation of verifiable controllable linkability would break backward unlinkability as verifiers would receive a proof that a specific signature links to a specific entry on RL (that must be publicly available in this case), which means that verifiers could link all signatures of revoked members. Although backward unlinkability can be achieved by using disjunctive zero-knowledge proofs (*i.e.*, OR proofs), such proofs also introduce non-trivial overhead for the involved entities. Thus, we suggest that RAs sign the returned response, similar to traditional OCSP responses. In order to reduce the communication and computational costs one could additionally employ distributed OCSP (D-OCSP) as proposed by Koga and Sakurai [37], such that all RAs have a different signing key although they share the same public key.

Security Model. In contrast to other revocation mechanisms, the RA in our setting only returns a boolean decision, *i.e.*, whether or not the signer has been revoked. Thus, unlike other revocation mechanisms, the RA does not reveal additional revocation-related information which would require to integrate the revocation feature into the formal security model of the GSS. Basically, LBR is an application of the feature of controllable linkability and, thus, we do not need to formally model our revocation mechanism in the security model. In fact, the role of the RA is already covered by the security model of group signatures with controllable linkability and our revocation mechanism is already considered in the original model (although this functionality is now implemented by the RA). Consequently, we also do not modify the security properties listed in Section 3.4. In addition, correctness of revocation follows from correctness of the GSS with controllable linkability.

6 Applying Linking-Based Revocation

For illustration purposes, we show how linking-based revocation can be applied to the *eXtremely Short Group Signature* scheme (XSGS) [21]. By means of the generic compiler [56], XSGS can be turned into a GSS with controllable linkability for free (without any overhead). Since LBR is transparent for signers, we only need to modify the GkGen algorithm. GkGen generates the additional linking key, which allows the revocation authority (in cooperation with multiple linking authorities) to perform the revocation check. Furthermore, we add the algorithms $\mathsf{CheckStatus}$ and Revoke according to the model for GSs with controllable linkability. The scheme is as follows (cf. [21] for more details):

$\mathsf{GkGen}(1^\lambda, t, n)$**:** The master opening key is $\mathsf{mok} = (\xi_1, \xi_2)$ for two randomly chosen elements $\xi_1, \xi_2 \in \mathbb{Z}_p$. The master linking key is $\mathsf{mlk} = (\hat{r}, \hat{s} = \hat{r}^{\xi_1})$ for a randomly chosen element $\hat{r} \in \mathbb{G}_2$. The master issuing key consists of $\mathsf{mik} = (\gamma, \hat{r})$ for a randomly chosen element $\gamma \in \mathbb{Z}_p$. The master linking key mlk is distributed among n linking authorities via $\mathsf{mlk}_i = (\hat{r}_i, \hat{s}_i) \leftarrow \mathsf{DKAut}(\mathsf{mlk}, t, n)$. The group public key is $\mathsf{gpk} = (g_1, k, h = k^{\xi_1}, g = k^{\xi_2}, \hat{g}_2, \hat{w} = \hat{g}_2^\gamma) \in \mathbb{G}_1^4 \times \mathbb{G}_2^2$.

$\mathsf{Issue}(\mathsf{gpk}, \mathsf{mik}, \mathbf{reg})$**:** The Issue algorithm interacts with the Join algorithm to add a user to the group. Thereby, a user receives a membership certificate (A_i, x, y), such that $A_i^{x+\gamma} = g_1 h^y$, where $y \in \mathbb{Z}_p$ is known to the user only. The issuing authority adds all the relevant information to the registration table $\mathbf{reg}$.

$\mathsf{GSig}(\mathsf{gpk}, M, \mathsf{gsk}_i)$**:** Given the group public key gpk, a message M, and a user's signing key $\mathsf{gsk}_i = (A, x, y) \in \mathbb{G}_1 \times \mathbb{Z}_p^2$, a signature is computed as follows. It randomly selects $\alpha, \beta \in \mathbb{Z}_p$ and encrypts the membership certificate:

$$T_1 = k^\alpha \qquad T_2 = Ah^\alpha \qquad T_3 = k^\beta \qquad T_4 = Ag^\beta$$

Then it sets $z = x\alpha + y$ and computes the non-interactive zero-knowledge proof of knowledge (α, β, x, z) as follows. It picks blinding values $r_\alpha, r_\beta, r_x, r_z \in \mathbb{Z}_p$ and computes the following values:

$$\begin{aligned}
&R_1 = k^{r_\alpha} \qquad R_3 = k^{r_\beta} \qquad R_4 = h^{r_\alpha}/g^{r_\beta} \\
&R_2 = e(T_2, \hat{g}_2)^{r_x} \cdot e(h, \hat{w})^{-r_\alpha} \cdot e(h, \hat{g}_2)^{-r_z} \\
&c = H(M, T_1, T_2, T_3, T_4, R_1, R_2, R_3, R_4) \\
&s_\alpha = r_\alpha + c\alpha \qquad s_\beta = r_\beta + c\beta \\
&s_x = r_x + cx \qquad s_z = r_z + cz
\end{aligned}$$

Finally, output the signature $\sigma = (T_1, T_2, T_3, T_4, c, s_\alpha,\ s_\beta, s_x, s_z)$.

$\mathsf{GVf}(\mathsf{gpk}, M, \sigma)$**:** Given the group public key gpk, a message M and a corresponding signature σ, verification is performed by checking the following relations:

$$\begin{aligned}
&k^{s_\alpha} = R_1 \cdot T_1^c \qquad k^{s_\beta} = R_3 \cdot T_3^c \qquad h^{s_\alpha}/g^{s_\beta} = R_4 \cdot \left(\tfrac{T_2}{T_4}\right)^c \\
&e(T_2, \hat{g}_2)^{s_x} \cdot e(h, \hat{w})^{-s_\alpha} \cdot e(h, \hat{g}_2)^{-s_z} = R_2 \cdot \left(\tfrac{e(g_1, \hat{g}_2)}{e(T_2, \hat{w})}\right)^c
\end{aligned}$$

If all of the above relations hold, the algorithm returns `true` and `false` otherwise.

$\mathsf{CheckStatus}(\mathsf{RL}, \mathcal{L}, \sigma)$**:** Given a revocation list RL consisting of revocation tokens of revoked members, a list of available linking authorities $\mathcal{L}$, and a signature σ, it determines the revocation status of the signer corresponding to the signature σ in question. Therefore, it interacts with a subset of at least t linking authorities $\mathcal{I} \subseteq \mathcal{L}$, *i.e.*, $|\mathcal{I}| \geq t$, to retrieve t comparison shares $\{C_i, D_i, \perp, \perp\} \leftarrow \mathsf{TShare}((T_1, T_2), \perp, \mathsf{mlk}_i)$ as follows:

$$C_i = e(T_2, \hat{r}_i) \qquad D_i = e(T_1, \hat{s}_i).$$

Based on these comparison shares, it computes the revocation token t via $\mathsf{t} = \mathsf{TSCom}(\{C_i, D_i, \perp, \perp\}_{i \in \mathcal{I}})$ as follows:

$$L_i = \prod_{j \in \mathcal{I}, j \neq i} \frac{j}{j - i} \qquad \mathsf{t} = \prod_{i \in \mathcal{I}} C_i^{L_i} \cdot \left(\prod_{i \in \mathcal{I}} D_i^{L_i}\right)^{-1}$$

Return `true` (revoked) if $\mathsf{t} \in \mathsf{RL}$ and `false` (not revoked) otherwise.

$\mathsf{Revoke}(\mathsf{gpk}, \mathsf{mik}, \mathbf{reg}, \mathsf{RL}, i)$**:** Given the group public key gpk, the master issuing key mik, the registration table $\mathbf{reg}$, the current revocation list RL, and a user i to be revoked[5], the algorithm computes the revocation token $\mathsf{t}_i = e(A_i, \hat{r})$ for user i and adds it to the revocation list, *i.e.*, $\mathsf{RL} = \mathsf{RL} \cup \{\mathsf{t}_i\}$.

[5] Again, revocation can also be done based on a user's signature $\sigma = (T, \pi)$ by means of mlk in which case the user's identity will not be required.

Security Analysis. We know that XSGS scheme is secure in the BSZ model (as shown in [21]) and in particular uses twin ElGamal encryption. We simply use its AoN-PKEET* version (according to the generic compiler [56]), which yields a secure group signatures with controllable linkability. Token indistinguishability, as defined and shown for standard ElGamal in Section 4.2, also holds for twin ElGamal and consequently an adversary cannot learn anything from the anonymous revocation tokens on RL.

7 Conclusion

It is well known that revocation mechanisms represent the major bottleneck in group signature schemes (see e.g., [41]). However, the general belief that any online authority must be prevented unnecessarily restricts the efficiency and practicality of revocation in group signature schemes. In this paper, we showed that the major drawbacks of existing revocation mechanisms, e.g., additional computations/updates for signers and verifiers, can be overcome by a paradigm-shift towards the incorporation of an online revocation authority. Since many applications and services already rely on always-connected devices that permanently interact with cloud computing infrastructures, the introduction of such an online revocation authority is absolutely reasonable. Considering (1) the significant performance gain (constant-time revocation checks), (2) the transparency for signers as well as verifiers, and (3) the general applicability to well-established PB-GSSs, we claim that our approach advances the open issue of efficient membership revocation in the context of GSSs. Hence, linking-based revocation represents a valuable contribution to the existing portfolio of revocation mechanisms.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. Daniel Slamanig has been supported by the H2020 project Prismacloud, grant agreement number 644962. Raphael Spreitzer and Thomas Unterluggauer have been supported by the European Commission through the FP7 program under project number 610436 (project MATTHEW).

References

1. G. Ateniese, D. X. Song, and G. Tsudik. Quasi-Efficient Revocation in Group Signatures. In *Financial Cryptography – FC 2002*, pages 183–197, 2002.
2. N. Attrapadung, K. Emura, G. Hanaoka, and Y. Sakai. A Revocable Group Signature Scheme from Identity-Based Revocation Techniques: Achieving Constant-Size Revocation List. In *Applied Cryptography and Network Security – ACNS 2014*, pages 419–437, 2014.
3. M. H. Au, P. P. Tsang, W. Susilo, and Y. Mu. Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems. In *Topics in Cryptology – CT-RSA 2009*, pages 295–308, 2009.

4. J. Baek and Y. Zheng. Identity-Based Threshold Decryption. In *Public Key Cryptography – PKC 2004*, pages 262–276, 2004.
5. M. Bellare, H. Shi, and C. Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. In *Topics in Cryptology – CT-RSA 2005*, pages 136–153, 2005.
6. O. Blazy, D. Derler, D. Slamanig, and R. Spreitzer. Non-Interactive Plaintext (In-)Equality Proofs and Group Signatures with Verifiable Controllable Linkability. In *Topics in Cryptology – CT-RSA 2016*, pages 127–143, 2016.
7. D. Boneh, X. Boyen, and H. Shacham. Short Group Signatures. In *Advances in Cryptology – CRYPTO 2004*, pages 41–55, 2004.
8. D. Boneh and H. Shacham. Group Signatures with Verifier-Local Revocation. In *Computer and Communications Security – CCS*, pages 168–177, 2004.
9. J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth. Foundations of Fully Dynamic Group Signatures. In *Applied Cryptography and Network Security – ACNS 2016*, pages 117–136, 2016.
10. E. Bresson and J. Stern. Efficient Revocation in Group Signatures. In *Public Key Cryptography – PKC 2001*, pages 190–206, 2001.
11. E. Brickell and J. Li. Enhanced Privacy ID from Bilinear Pairing for Hardware Authentication and Attestation. In *Social Computing – SocialCom / Privacy, Security, Risk and Trust – PASSAT 2010*, pages 768–775, 2010.
12. E. Brickell and J. Li. Enhanced Privacy ID: A Direct Anonymous Attestation Scheme with Enhanced Revocation Capabilities. *IEEE Trans. Dependable Sec. Comput.*, 9:345–360, 2012.
13. J. Camenisch and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *Advances in Cryptology – CRYPTO 2002*, pages 61–76, 2002.
14. J. Camenisch and M. Stadler. Efficient Group Signature Schemes for Large Groups. In *Advances in Cryptology – CRYPTO 1997*, pages 410–424, 1997.
15. S. Canard, I. Coisel, G. de Meulenaer, and O. Pereira. Group Signatures are Suitable for Constrained Devices. In *Information Security and Cryptology – ICISC 2010*, pages 133–150, 2010.
16. S. Canard, N. Desmoulins, J. Devigne, and J. Traoré. On the Implementation of a Pairing-Based Cryptographic Protocol in a Constrained Device. In *Pairing-Based Cryptography – Pairing 2012*, pages 210–217, 2012.
17. D. Chaum and E. van Heyst. Group Signatures. In *Advances in Cryptology – EUROCRYPT 1991*, pages 257–265, 1991.
18. S. S. M. Chow, W. Susilo, and T. H. Yuen. Escrowed Linkability of Ring Signatures and Its Applications. In *Progress in Cryptology – VIETCRYPT 2006*, pages 175–192, 2006.
19. D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, RFC Editor, May 2008. http://www.rfc-editor.org/rfc/rfc5280.txt.
20. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *CRYPTO*, pages 13–25, 1998.
21. C. Delerablée and D. Pointcheval. Dynamic Fully Anonymous Short Group Signatures. In *Progress in Cryptology – VIETCRYPT 2006*, pages 193–210, 2006.
22. K. Emura and T. Hayashi. A Light-Weight Group Signature Scheme with Time-Token Dependent Linking. In *Lightweight Cryptography for Security and Privacy – LightSec 2015*, pages 37–57, 2015.

23. K. Emura, A. Miyaji, and K. Omote. An r-Hiding Revocable Group Signature Scheme: Group Signatures with the Property of Hiding the Number of Revoked Users. *J. Applied Mathematics*, 2014:983040:1–983040:14, 2014.
24. C. Fan, R. Hsu, and M. Manulis. Group Signature with Constant Revocation Costs for Signers and Verifiers. In *Cryptology and Network Security – CANS 2011*, pages 214–233, 2011.
25. P. Fouque and D. Pointcheval. Threshold Cryptosystems Secure against Chosen-Ciphertext Attacks. In *Advances in Cryptology – ASIACRYPT 2001*, pages 351–368, 2001.
26. E. Ghadafi. Efficient Distributed Tag-Based Encryption and Its Application to Group Signatures with Efficient Distributed Traceability. In *Progress in Cryptology – LATINCRYPT 2014*, pages 327–347, 2014.
27. G. Grewal, R. Azarderakhsh, P. Longa, S. Hu, and D. Jao. Efficient Implementation of Bilinear Pairings on ARM Processors. In *Selected Areas in Cryptography – SAC 2012*, pages 149–165, 2012.
28. J. Groth and A. Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In *Advances in Cryptology - EUROCRYPT 2008*, pages 415–432, 2008.
29. J. Y. Hwang, L. Chen, H. S. Cho, and D. Nyang. Short Dynamic Group Signature Scheme Supporting Controllable Linkability. *IEEE Trans. Information Forensics and Security*, 10:1109–1124, 2015.
30. J. Y. Hwang, S. Lee, B. Chung, H. S. Cho, and D. Nyang. Group Signatures with Controllable Linkability for Dynamic Membership. *Information Sciences*, 222:761–778, 2013.
31. J. Y. Hwang, S. Lee, B.-H. Chung, H. S. Cho, and D. Nyang. Short Group Signatures with Controllable Linkability. In *LightSec*, pages 44–52. IEEE, 2011.
32. International Organization for Standardization (ISO). ISO/IEC 20008-2: Information technology – Security techniques – Anonymous digital signatures – Part 2: Mechanisms using a group public key, November 2013.
33. A. P. Isern-Deyà, L. Huguet-Rotger, M. Payeras-Capellà, and M. Mut-Puigserver. On the Practicability of Using Group Signatures on Mobile Devices: Implementation and Performance Analysis on the Android Platform. *International Journal of Information Security*, 14:335–345, 2015.
34. A. P. Isern-Deyà, A. Vives-Guasch, M. M. Puigserver, M. Payeras-Capellà, and J. Castellà-Roca. A Secure Automatic Fare Collection System for Time-Based or Distance-Based Services with Revocable Anonymity for Users. *The Computer Journal*, 56:1198–1215, 2013.
35. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable Signatures. In *Advances in Cryptology – EUROCRYPT 2004*, pages 571–589, 2004.
36. A. Kiayias and M. Yung. Group Signatures with Efficient Concurrent Join. In *Advances in Cryptology – EUROCRYPT 2005*, pages 198–214, 2005.
37. S. Koga and K. Sakurai. A Distributed Online Certificate Status Protocol with a Single Public Key. In *Public Key Cryptography – PKC 2004*, pages 389–401, 2004.
38. V. Kumar, H. Li, J. J. Park, K. Bian, and Y. Yang. Group Signatures with Probabilistic Revocation: A Computationally-Scalable Approach for Providing Privacy-Preserving Authentication. In *Computer and Communications Security – CCS 2015*, pages 1334–1345, 2015.
39. B. Libert, T. Peters, and M. Yung. Group Signatures with Almost-for-Free Revocation. In *Advances in Cryptology – CRYPTO 2012*, pages 571–589, 2012.
40. B. Libert, T. Peters, and M. Yung. Scalable Group Signatures with Revocation. In *Advances in Cryptology – EUROCRYPT 2012*, pages 609–627, 2012.

41. M. Manulis, N. Fleischhacker, F. K. Felix Günther, and B. Poettering. Group Signatures: Authentication with Privacy. Technical report, BSI – Federal Office for Information Security, 2012.
42. T. Nakanishi, H. Fujii, Y. Hira, and N. Funabiki. Revocable Group Signature Schemes with Constant Costs for Signing and Verifying. In *Public Key Cryptography – PKC 2009*, pages 463–480, 2009.
43. T. Nakanishi, T. Fujiwara, and H. Watanabe. A Linkable Group Signature and Its Application to Secret Voting. *Trans. of Information Processing Society of Japan*, 40(7), 1999.
44. T. Nakanishi and N. Funabiki. Verifier-Local Revocation Group Signature Schemes with Backward Unlinkability from Bilinear Maps. In *Advances in Cryptology – ASIACRYPT 2005*, pages 533–548, 2005.
45. T. Nakanishi and N. Funabiki. A Short Verifier-Local Revocation Group Signature Scheme with Backward Unlinkability. In *International Workshop on Security – IWSEC 2006*, pages 17–32, 2006.
46. T. Nakanishi, F. Kubooka, N. Hamada, and N. Funabiki. Group Signature Schemes with Membership Revocation for Large Groups. In *Information Security and Privacy – ACISP 2005*, pages 443–454, 2005.
47. T. Nakanishi and Y. Sugiyama. A Group Signature Scheme with Efficient Membership Revocation for Reasonable Groups. In *Information Security and Privacy – ACISP 2004*, pages 336–347, 2004.
48. M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *Symposium on the Theory of Computing – STOC*, pages 427–437, 1990.
49. L. Nguyen and R. Safavi-Naini. Efficient and Provably Secure Trapdoor-Free Group Signature Schemes from Bilinear Pairings. In *Advances in Cryptology – ASIACRYPT 2004*, pages 372–386, 2004.
50. R. Pagh and F. F. Rodler. Cuckoo Hashing. *Journal of Algorithms*, 51:122–144, 2004.
51. Ponemon Institute LLC. 2015 PKI Global Trends Study, 2015.
52. K. Potzmader, J. Winter, D. M. Hein, C. Hanser, P. Teufl, and L. Chen. Group Signatures on Mobile Devices: Practical Experiences. In *Trust and Trustworthy Computing – TRUST 2013*, pages 47–64, 2013.
53. C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Advances in Cryptology - CRYPTO '91*, pages 433–444, 1991.
54. S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP. RFC 6960, Internet Engineering Task Force (IETF), June 2013. https://www.ietf.org/rfc/rfc6960.txt.
55. A. Shamir. How to Share a Secret. *Communications of the ACM*, 22:612–613, 1979.
56. D. Slamanig, R. Spreitzer, and T. Unterluggauer. Adding Controllable Linkability to Pairing-Based Group Signatures for Free. In *Information Security – ISC 2014*, pages 388–400, 2014.
57. Q. Tang. Public Key Encryption Supporting Plaintext Equality Test and User-Specified Authorization. *Security and Communication Networks*, 5:1351–1362, 2012.
58. I. Teranishi, J. Furukawa, and K. Sako. k-Times Anonymous Authentication (Extended Abstract). In *Advances in Cryptology – ASIACRYPT 2004*, pages 308–322, 2004.

59. T. Unterluggauer and E. Wenger. Efficient Pairings and ECC for Embedded Systems. In *Cryptographic Hardware and Embedded Systems – CHES 2014*, pages 298–315, 2014.
60. E. R. Verheul. Practical Backward Unlinkable Revocation in FIDO, German e-ID, Idemix and U-Prove. *IACR Cryptology ePrint Archive 2016/217*, 2016.
61. V. K. Wei. Tracing-by-Linking Group Signatures. In *Information Security – ISC 2005*, pages 149–163, 2005.
62. S. Zhou and D. Lin. Shorter Verifier-Local Revocation Group Signatures from Bilinear Maps. In *Cryptology and Network Security – CANS 2006*, pages 126–143, 2006.