Non-Interactive Plaintext (In-)Equality Proofs and Group Signatures with Verifiable Controllable Linkability

Olivier Blazy, David Derler, Daniel Slamanig, Raphael Spreitzer

Non-Interactive Plaintext (In-)Equality Proofs and Group Signatures with Verifiable Controllable Linkability

Olivier Blazy[1], David Derler[2,‡], Daniel Slamanig[2,‡], and Raphael Spreitzer[2,§]

[1] Université de Limoges, XLim, France
olivier.blazy@unilim.fr
[2] IAIK, Graz University of Technology, Austria
{david.derler|daniel.slamanig|raphael.spreitzer}@tugraz.at

Abstract. Group signatures are an important privacy-enhancing tool that allow to anonymously sign messages on behalf of a group. A recent feature for group signatures is controllable linkability, where a dedicated linking authority (LA) can determine whether two given signatures stem from the same signer without being able to identify the signer(s). Currently the linking authority is fully trusted, which is often not desirable.

In this paper, we firstly introduce a generic technique for non-interactive zero-knowledge plaintext equality and inequality proofs. In our setting, the prover is given two ciphertexts and some trapdoor information, but neither has access to the decryption key nor the randomness used to produce the respective ciphertexts. Thus, the prover performs these proofs on *unknown* plaintexts. Besides a generic technique, we also propose an efficient instantiation that adapts recent results from Blazy et al. (CT-RSA'15), and in particular a combination of Groth-Sahai (GS) proofs (or sigma proofs) and smooth projective hash functions (SPHFs).

While this result may be of independent interest, we use it to realize verifiable controllable linkability for group signatures. Here, the LA is required to non-interactively prove whether or not two signatures link (while it is not able to identify the signers). This significantly reduces the required trust in the linking authority. Moreover, we extend the model of group signatures to cover the feature of verifiable controllable linkability.

1 Introduction

Group signatures, introduced by Chaum and van Heyst [CvH91], allow users to anonymously sign messages on behalf of a group. In case of dispute, a so-called opening authority is able to reveal the identity of the actual signer. While many popular group signature schemes (GSSs) (such as [ACJT00,BBS04]) simply trust

This is the full version of a paper to appear at CT-RSA 2016.

‡ Supported by EU H2020 project PRISMACLOUD, grant agreement n°644962.
§ Supported by the Austrian Research Promotion Agency (FFG) and the Styrian Business Promotion Agency (SFG), grant agreement n°836628 (SECOS).

the output of the opening authority, Camenisch and Stadler [CS97] proposed to require a proof of the correctness of the opening mechanism. Later, Bellare et al. [BSZ05] introduced a model for dynamic group signatures (BSZ model) that incorporates this issue by requiring *publicly verifiable proofs of opening*, i.e., the opening authority provides a proof that the claimed signer indeed produced a given signature. Recently, Sakai et al. [SSE+12] identified an issue with this opening mechanism in the BSZ model and introduced an additional property called opening soundness. This property prevents signature hijacking, i.e., it prevents malicious group members (who cooperate with the opening authority) from claiming ownership of a signature produced by an honest group member. Over the years many other additional features for GSSs have been introduced (cf. Section 1.2).

One rather recent feature is called controllable linkability [HLhC+11,HLC+13, HCCN15,SSU14]. Here, a dedicated entity called linking authority (LA) can determine whether two given group signatures stem from the same signer, but the LA is *not* able to identify the signer(s). Consequently, the LA is strictly less powerful than the opening authority which can identify all signers by opening their signatures. Like early group signatures did not consider untrusted opening authorities, existing group signatures with controllable linkability [HLhC+11, HLC+13,HCCN15,SSU14] do *not* consider untrusted LAs. In particular, the LA simply provides a binary linking decision and thus has to be fully trusted. It is, however, desirable to reduce this trust. Ideally, in a way that the LA needs to provide verifiable evidence, i.e., a proof, of a correct decision. In this paper, we solve this open problem and introduce the novel concept of *verifiable controllable linkability* (VCL). Applications of VCL include different types of privacy-preserving data-mining scenarios in various fields such as online shopping, public transport, park- and road pricing. Essentially, whenever one requires to analyse customers' behavioural patterns in a privacy-respecting way and these computations are outsourced to a potentially untrusted party, e.g., a cloud provider, that needs to prove honest behaviour and must not be able to identify individuals. Moreover, their application to revocation mechanisms seems interesting to study.

1.1 Background and Motivation

Naive approaches to solve this problem, like abusing the opening authority or requiring the LA to sign its decision, are rather privacy intrusive and/or not satisfactory. To give an idea of how we approach this problem, we have to look at the existing approaches to achieve controllable linkability without verifiability. This concept has been proposed for several GSSs by Hwang et al. [HLhC+11, HLC+13, HCCN15]. As their approach to controllable linkability, however, is ad-hoc and always tailored to a specific GSS, Slamanig et al. [SSU14] proposed a generic approach to add controllable linkability to pairing-based group signature schemes following the sign-and-encrypt-and-prove (SEP) paradigm (cf. Section 2.3), which covers a large class of practical group signatures in the ROM. We recall that a group signature in the SEP paradigm is an encryption of a per-user unique value (certificate) under the public key of the

opening authority and a non-interactive zero-knowledge proof of a signature (on this certificate) from the group manager. This generic approach allows the LA to perform the linking operation on the encrypted membership certificates (which are used for opening group signatures) by means of a variant of the all-or-nothing public key encryption with equality tests (AoN-PKeet*) primitive. Basically, the LA obtains a *single* linking key (trapdoor) that allows plaintext equality tests on the membership certificates without being able to decrypt. Now, our idea is to require the LA to provide a proof that either two encrypted membership certificates contain the same or different unknown certificates (plaintexts). The particular challenge, however, is that the LA must not be able to identify the signers and thus needs to perform such proofs without knowing the plaintexts, the decryption key or the randomness used to produce the ciphertexts. Moreover, in contrast to opening proofs, we do not only need to provide a proof in case of a positive linking decision but also in case of a *negative* decision, i.e., when two ciphertexts contain different unknown plaintexts (certificates). This makes proving the correctness of a linking decision a much more challenging task.

1.2 Related Work

Group signatures. In traceable signatures [KTY04, Cho09], the opening authority can compute a tracing trapdoor for a user, which allows the identification of all signatures generated by a particular user without violating the privacy of other users. In group signatures with message dependent opening [SEH+12], the opening authority cannot open any signature unless an additional authority (the admitter) admits to open signatures for specified messages and thus restricts the power of the opening authority. In deniable group signatures [IEH+15], the opener can, in addition to opening proofs, prove that a particular signature has not been generated by a particular signer. Apart from these opening capabilities, also linking capabilities have been investigated. For instance, the possibility to publicly link group signatures of users without identifying them [NFW99] or to allow public tracing of signers who have produced a number of signatures above a certain threshold [Wei05]. But also the linkability of signatures for a specified time frame (by fixing the randomness for a certain time [MCVH12] or by introducing specific time tokens [EH14]) have been considered. Another direction is to put the user in charge of controlling which signatures can be linked, as it is used in DAA [BCC04] and related schemes [BFG+13]. These concepts are related to our work but do not help to realize our goals.

Plaintext equality/inequality proofs. Zero-knowledge proofs of plaintext equality (under distinct public keys) are well known from the twin-encryption paradigm [NY90]. However, we require equality as well as inequality proofs and in our setting the prover neither has access to the decryption key nor the randomness used to produce the respective ciphertexts. Jakobsson and Juels [JJ00] introduced the concept of distributed plaintext equality tests (PETs) within their approach to general secure multiparty computation. Basically, it allows

$n > 1$ entities to determine whether two ElGamal ciphertexts encrypt the same or a different message without learning the message. However, this requires access to the decryption key. Choi et al. [CEJ+07] provide zero-knowledge equality/inequality proofs for boolean ElGamal ciphertexts. Their approach requires the knowledge of the decryption key and the randomness used to produce the two ciphertexts. Parkes et al. [PRST08] provide zero-knowledge equality/inequality proofs of plaintexts within Paillier ciphertexts, which however require either access to the randomness used to produce the ciphertexts or access to the plaintexts. Recently, Blazy et al. [BCV15] introduced a generic approach to prove non-membership with respect to some language in non-interactive zero-knowledge. Among others, they show how to prove plaintext inequality of two ElGamal ciphertexts, where the verifier knows the plaintext and the randomness used to produce one of the ciphertexts. Therefore, none of these approaches directly fits our requirements.

1.3 Contribution

The contributions of this paper are as follows: (1) Based upon the idea of public key encryption with equality tests, we define a generic non-interactive proof system that allows to perform zero-knowledge proofs about plaintext equality and inequality with respect to any two ciphertexts under the same public key. Thereby, the prover is neither required to have access to the decryption key nor to the randomness used to produce the respective ciphertexts. (2) We show how Groth-Sahai (GS) proofs [GS08] and an adaptation of *non-interactive zero-knowledge proofs of non-membership* [BCV15] can be combined to obtain an instantiation of our proof system. While an instantiation of such a proof system is of independent interest, it allows us to construct group signatures with *verifiable controllable linkability* (VCL-GS). (3) We adopt the model of GSSs with controllable linkability [HLhC+11, HLC+13, HCCN15] to a model for VCL-GS. In the vein of Sakai et al. [SSE+12], we introduce a property called *linking soundness*, which requires that even corrupted LAs (colluding with malicious users) cannot produce false linking proofs. (4) We show how to transform GSSs with controllable linkability following the SEP paradigm into GSSs with verifiable controllable linkability by using the proposed non-interactive zero-knowledge proof system.

2 Preliminaries

Subsequently, we discuss preliminaries and recall assumptions and required tools.

Notation. Let $x \xleftarrow{R} X$ denote the operation that picks an element x uniformly at random from a set X. A function $\epsilon : \mathbb{N} \to \mathbb{R}^+$ is called negligible if for all $c > 0$ there is a k_0 such that $\epsilon(k) < 1/k^c$ for all $k > k_0$. In the remainder of this paper, we use ϵ to denote such a negligible function. We use boldface letters to denote vectors, e.g., $\mathbf{X} = (X_1, \ldots X_n)$.

Let $\mathbb{G}_1 = \langle g \rangle$, $\mathbb{G}_2 = \langle \hat{g} \rangle$, and $\mathbb{G}_T$ be groups of prime order p. We write elements in $\mathbb{G}_2$ as $\hat{g}, \hat{h}$, etc. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map, where it holds for all $(u, \hat{v}, a, b) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{Z}_p^2$ that $e(u^a, \hat{v}^b) = e(u, \hat{v})^{ab}$, and $e(g, \hat{g}) \neq 1$, and e is efficiently computable. We assume the asymmetric setting where $\mathbb{G}_1 \neq \mathbb{G}_2$. The required hardness assumptions are provided in Appendix A.

2.1 Groth-Sahai (GS) Non-Interactive Zero-Knowledge Proofs

Groth and Sahai [GS08] provide a framework for efficient non-interactive witness-indistinguishable (NIWI) and non-interactive zero-knowledge (NIZK) proofs for languages defined over bilinear groups. It allows, among others, to prove statements about the satisfiability of so-called pairing product equations (PPEs). While the framework is quite independent of the underlying hardness assumption, we will use the instantiation based on the SXDH setting, and, thus, our further explanations are tailored to this setting. A PPE is of the form

$$\prod_{i=1}^{n} e(A_i, \hat{Y}_i) \cdot \prod_{i=1}^{m} e(X_i, \hat{B}_i) \cdot \prod_{i=1}^{m} \prod_{j=1}^{n} e(X_i, \hat{Y}_j)^{\gamma_{ij}} = t_T \ ,$$

where $\mathbf{X} \in \mathbb{G}_1^m$, $\hat{\mathbf{Y}} \in \mathbb{G}_2^n$ are the secret vectors (to prove knowledge of) and $\mathbf{A} \in \mathbb{G}_1^n, \hat{\mathbf{B}} \in \mathbb{G}_2^m, \Gamma = (\gamma_{ij})_{i \in [m], j \in [n]} \in \mathbb{Z}_p^{n \cdot m}$, and $t_T \in \mathbb{G}_T$ are public constants. Informally, GS proofs use the following strategy. One commits to the vectors $\mathbf{X}$ and $\hat{\mathbf{Y}}$, and uses the commitments instead of the actual values in the PPE. The proof π is used to cancel out the randomness used in the commitments. As this does not directly work when using the groups $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$, one projects the involved elements to the vector spaces $\mathbb{G}_1^2, \mathbb{G}_2^2$, and $\mathbb{G}_T^4$ by using the defined projection maps and proves the satisfiability of the PPE using the projected elements and corresponding bilinear map $F : \mathbb{G}_1^2 \times \mathbb{G}_2^2 \to \mathbb{G}_T^4$.

More formally, a GS proof for a PPE allows to prove knowledge of a witness $w = (\mathbf{X}, \hat{\mathbf{Y}})$ such that the PPE, uniquely defined by the statement $x = (\mathbf{A}, \hat{\mathbf{B}}, \Gamma, t_T)$, is satisfied. Henceforth, let BG denote the description of the used bilinear group and let R be the relation such that $(\mathsf{BG}, x, w) \in R$ iff w is a satisfying witness for x with respect to BG. Further, let L_R be the corresponding language.

Formally, a non-interactive proof system in a bilinear group setting is defined as follows:

Definition 1. *A non-interactive proof system Π is a tuple of PPT algorithms* $(\mathsf{BGGen}, \mathsf{CRSGen}, \mathsf{Proof}, \mathsf{Verify})$*, which are defined as follows:*

$\mathsf{BGGen}(1^\kappa)$: Takes a security parameter κ as input, and outputs a bilinear group description BG.

$\mathsf{CRSGen}(\mathsf{BG})$: Takes a bilinear group description BG as input, and outputs a common reference string crs.

$\mathsf{Proof}(\mathsf{BG}, \mathsf{crs}, x, w)$: Takes a bilinear group description BG, a common reference string crs, a statement x, and a witness w as input, and outputs a proof π.

$\mathsf{Verify}(\mathsf{BG}, \mathsf{crs}, x, \pi)$: Takes a bilinear group description BG, a common reference string crs, a statement x, and a proof π as input, and outputs 1 if π is valid and 0 otherwise.

The security definitions for non-interactive proof systems are provided in Appendix B. GS proofs are perfectly complete, perfectly sound, and witness indistinguishable. Furthermore, they are composably zero-knowledge if $t_T = 1_{\mathbb{G}_T}$ and the PPE does not involve a pairing of two public constants.

Throughout this paper we use the GS-based commit-and-prove approach from [EG14], which allows to reuse the commitments in proofs for different statements. This allows us to prove statements with respect to commitments that are included in the common reference string (CRS) to obtain more efficient proofs. Moreover, the fact that the commitments are already contained in the CRS allows us to exclude the usage of trivial witnesses, i.e., $1_{\mathbb{G}_1}$ or $1_{\mathbb{G}_2}$.

2.2 Smooth Projective Hash Functions

Smooth projective hash functions (SPHF) [CS02] are families of pairs of functions $(\mathsf{Hash}, \mathsf{ProjHash})$ defined on a language L. They are indexed by a pair of associated keys (hk, hp), where the hashing key hk may be viewed as the private key and the projection key hp as the public key. On a word $W \in L$, both functions need to yield the same result, i.e., $\mathsf{Hash}(hk, L, W) = \mathsf{ProjHash}(hp, L, W, w)$, where the latter evaluation additionally requires a witness w that $W \in L$. Thus, they can be seen as a tool for implicit designated-verifier proofs of membership [ACP09]. Formally SPHFs are defined as follows (cf. [BBC+13b]).

Definition 2. *A SPHF for a language L is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{Hash}, \mathsf{ProjHash})$*, which are defined as follows:*

$\mathsf{Setup}(1^\kappa)$: Takes a security parameter κ and generates the global parameters pp (we assume that all algorithms have access to pp).
$\mathsf{HashKG}(L)$: Takes a language L and outputs a hashing key hk for L.
$\mathsf{ProjKG}(hk, L, W)$: Takes a hashing key hk, a language L, and a word W and outputs a projection key hp, possibly depending on W.
$\mathsf{Hash}(hk, L, W)$: Takes a hashing key hk, a language L, and a word W and outputs a hash H'.
$\mathsf{ProjHash}(hp, L, W, w)$: Takes a projection key hp, a language L, a word W, and a witness w for $W \in L$ and outputs a hash H.

The security properties as well as the concrete ElGamal-based instantiation from [GL03] used in this paper are provided in Appendix C.

2.3 Sign-and-Encrypt-and-Prove Paradigm

Group signature schemes following the sign-and-encrypt-and-prove (SEP) paradigm are popular and there are various efficient constructions (in the ROM)

following this paradigm. Such a scheme consists of the following three building blocks: (1) A secure signature scheme $\mathcal{DS} = (\mathsf{KeyGen_s}, \mathsf{Sign}, \mathsf{Vrfy})$, (2) an at least IND-CPA secure public key encryption scheme $\mathcal{AE} = (\mathsf{KeyGen_e}, \mathsf{Enc}, \mathsf{Dec})$ and (3) a non-interactive zero-knowledge proof of knowledge (NIZKPK) system, e.g., non-interactive versions of Σ-protocols obtained via the Fiat-Shamir transform in the ROM (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 $\mathsf{cert} \leftarrow \mathsf{Sign}(\mathsf{sk}_s, f(x_i))$ which represents 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:

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

We note that there are slight deviations in instantiations of this paradigm (cf. [NFHF09, SSU14]), 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 $g(\cdot)$. We, however, stress that for our approach in this paper it does not matter how T is exactly constructed (beyond being the encryption of a per-user unique value).

2.4 All-or-Nothing Public Key Encryption With Equality Tests

Following the work of Tang [Tan12a, Tan12b], Slamanig et al. [SSU14] modified the concept of all-or-nothing public key encryption with equality tests (AoN-PKeet*). The idea of AoN-PKeet [Tan12a, Tan12b] is to allow specific entities in possession of a trapdoor to perform equality tests on ciphertexts without learning the underlying plaintexts. Slamanig et al. additionally require this primitive to be compatible with efficient zero-knowledge proofs regarding the plaintexts, to ensure compatibility with GSSs following the SEP paradigm.

An AoN-PKeet* scheme (KeyGen, Enc, Dec, Aut, Com) is a conventional (at least IND-CPA secure) public key encryption scheme (compatible with efficient zero-knowledge proofs) augmented by two additional algorithms Aut and Com (cf. [SSU14] for a formal treatment).

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

$\mathsf{Com}(T, T', \mathsf{tk})$: Takes two ciphertexts (T, T') and a trapdoor tk and returns 1 if both ciphertexts encrypt the same (unknown) message and 0 otherwise.

Definition 3 ([SSU14]). *An* AoN-PKeet* *scheme is called 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-CCA security against Type-II adversaries (outsiders).*

Construction from ElGamal. In a bilinear group setting where the (S)XDH assumption is assumed to hold, one can rely on ElGamal encryption in $\mathbb{G}_1$. Let the private key be a random element $\xi \xleftarrow{R} \mathbb{Z}_p$ and the corresponding public key be $h \leftarrow g^{\xi} \in \mathbb{G}_1$, then the encryption of a message m is computed as $T = (T_1, T_2) = (g^{\alpha}, mh^{\alpha})$ for a randomly chosen element $\alpha \xleftarrow{R} \mathbb{Z}_p$. The trapdoor generation and comparison algorithms are as follows:

$\mathsf{Aut}(\xi)$: Return the trapdoor $\mathsf{tk} \leftarrow (\hat{r}, \hat{t} = \hat{r}^{\xi}) \in \mathbb{G}_2^2$ for a random $\hat{r} \xleftarrow{R} \mathbb{G}_2$.
$\mathsf{Com}(T, T', \mathsf{tk})$: Given two ciphertexts $T = (T_1, T_2) = (g^{\alpha}, mh^{\alpha})$ and $T' = (T_1', T_2') = (g^{\alpha'}, m'h^{\alpha'})$ and a trapdoor $\mathsf{tk} = (\hat{r}, \hat{t} = \hat{r}^{\xi})$, return 1 if $e(T_2, \hat{r}) \cdot e(T_1, \hat{t})^{-1} = e(T_2', \hat{r}) \cdot e(T_1', \hat{t})^{-1}$ holds and 0 otherwise.

Lemma 1 ([SSU14]). *Under the co-CDH assumption* AoN-PKeet* *based on ElGamal in* $\mathbb{G}_1$ *in an (S)XDH setting is secure.*

3 Non-Interactive Plaintext (In-)Equality Proofs

We are interested in plaintext equality and inequality proofs where the prover neither knows the randomness used for encryption, nor the decryption key and consequently also does *not* know the plaintexts. By employing the idea of AoN-PKeet* [SSU14], the prover can use a trapdoor to determine whether two ciphertexts encrypt the same unknown plaintext, while not being able to decrypt. This, in turn, allows the prover to select which type of proof to conduct. Moreover, for AoN-PKeet* schemes in the pairing setting, we can use the pairing product equation that is used by the Com algorithm and a suitable proof framework to prove (1) knowledge of a trapdoor that is consistent with the respective public key and (2) the satisfiability of the pairing product equation corresponding to Com when used with the non-revealed trapdoor on two ciphertexts in question. As we will see later, this allows us to prove plaintext equality in a straightforward way, while plaintext inequality requires a slightly more sophisticated approach.

3.1 A Generic Construction

Let $\mathcal{PKEQ} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Aut}, \mathsf{Com})$ be a secure AoN-PKeet* scheme. Building upon $\mathcal{PKEQ}$, we define a generic non-interactive proof system Π that—for two ciphertexts T and T' under some public key pk—allows to prove knowledge of a trapdoor tk that either attests membership of (T, T', pk) in a language $L_{R_\in}$ or in a language $L_{R_\notin}$. The corresponding NP-relations are defined as follows:

$$((T, T', \mathsf{pk}), \mathsf{tk}) \in R_\in \iff \mathsf{Com}(T, T', \mathsf{tk}) = 1 \ \wedge \ \mathsf{tk} \equiv \mathsf{pk},$$
$$((T, T', \mathsf{pk}), \mathsf{tk}) \in R_\notin \iff \mathsf{Com}(T, T', \mathsf{tk}) = 0 \ \wedge \ \mathsf{tk} \equiv \mathsf{pk},$$

where $\mathsf{tk} \equiv \mathsf{pk}$ denotes that tk corresponds to pk and we omit BG for simplicity. To obtain a non-interactive proof system Π with the desired expressiveness, we compose two non-interactive proof systems, namely $\Pi_\in$ and $\Pi_\notin$. Here, $\Pi_\in$

covers statements in $L_{R_\in}$, whereas $\Pi_\notin$ covers statements in $L_{R_\notin}$. It is easy to see that—by the soundness of $\mathcal{PKEQ}$—each tuple $((T, T', \mathsf{pk}), \mathsf{tk})$ is either in $R_\in$ or in $R_\notin$. Membership can be efficiently checked using the Com algorithm. The non-interactive proof system Π is presented in Scheme 1, where we assume that one can efficiently decide for which language a given proof π has been computed.[1]

$\mathsf{BGGen}(1^\kappa)$: Takes a security parameter κ as input, runs $\mathsf{BG} \leftarrow \Pi_{\in,\notin}.\mathsf{BGGen}(1^\kappa)$ and returns BG.[a]

$\mathsf{CRSGen}(\mathsf{BG})$: Takes a bilinear group description BG as input, runs $\mathsf{crs}_\in \leftarrow \Pi_\in.\mathsf{CRSGen}(\mathsf{BG})$, and $\mathsf{crs}_\notin \leftarrow \Pi_\notin.\mathsf{CRSGen}(\mathsf{BG})$, and outputs a common reference string $\mathsf{crs} \leftarrow (\mathsf{crs}_\in, \mathsf{crs}_\notin)$.

$\mathsf{Proof}(\mathsf{BG}, \mathsf{crs}, (T, T', \mathsf{pk}), \mathsf{tk})$: Takes a bilinear group description BG, a common reference string crs, a statement (T, T', pk), and a witness tk. If $((T, T', \mathsf{pk}), \mathsf{tk}) \in R_\in$, return $\pi_\in \leftarrow \Pi_\in.\mathsf{Proof}(\mathsf{BG}, \mathsf{crs}_\in, (T, T', \mathsf{pk}), \mathsf{tk})$. Otherwise, return $\pi_\notin \leftarrow \Pi_\notin.\mathsf{Proof}(\mathsf{BG}, \mathsf{crs}_\notin, (T, T', \mathsf{pk}), \mathsf{tk})$.

$\mathsf{Verify}(\mathsf{BG}, \mathsf{crs}, (T, T', \mathsf{pk}), \pi)$: Takes a bilinear group description BG, a common reference string crs, a statement (T, T', pk) and a proof π. If π is for language $L_{R_\in}$ return $\Pi_\in.\mathsf{Verify}(\mathsf{BG}, \mathsf{crs}_\in, (T, T', \mathsf{pk}), \pi)$ and if π is for language $L_{R_\notin}$ return $\Pi_\notin.\mathsf{Verify}(\mathsf{BG}, \mathsf{crs}_\notin, (T, T', \mathsf{pk}), \pi)$.

[a] With $\Pi_{\in,\notin}$ we denote that both proof systems are with respect to the same bilinear group description.

Scheme 1: NIPEI Proof System

We call a non-interactive plaintext equality and inequality (NIPEI) proof system *secure* if it is perfectly complete, perfectly sound, and at least computationally zero-knowledge. The subsequent lemma trivially follows from the fact that $L_{R_\in}$ and $L_{R_\notin}$ are disjoint.

Lemma 2. *If $\Pi_\in$ and $\Pi_\notin$ are secure NIZK proof systems, then the resulting NIPEI proof system Π is also secure. Thereby, for every security property $p_\in$ of $\Pi_\in$ and corresponding security property $p_\notin$ of $\Pi_\notin$, Π inherits $p_\in$ if $p_\in$ is implied by $p_\notin$ and $p_\notin$ otherwise. That is, Π inherits the weaker security notion of both.*

3.2 Instantiation with $\mathcal{PKEQ}$ From ElGamal Encryption

We will now present a concrete instantiation of a NIPEI proof system in the SXDH setting where the $\mathcal{PKEQ}$ scheme is based on ElGamal encryption in $\mathbb{G}_1$. Recall that the public key is $\mathsf{pk} = g^\xi \in \mathbb{G}_1$, the trapdoor is $\mathsf{tk} = (\hat{r}, \hat{t} = \hat{r}^\xi) \in \mathbb{G}_2^2$ and for two ciphertexts T and T', $\mathsf{Com}(T, T', \mathsf{tk})$ checks whether $e(T_2, \hat{r}) \cdot e(T_1, \hat{t})^{-1} = e(T_2', \hat{r}) \cdot e(T_1', \hat{t})^{-1}$ holds. If so, the ciphertexts encrypt the same plaintexts and different plaintexts otherwise. Subsequently, we present the relations $R_\in$ and $R_\notin$ for this $\mathcal{PKEQ}$ scheme. For membership in $R_\in$, the

[1] As $L_{R_\in}$ and $L_{R_\notin}$ are disjoint, one can otherwise just run Verify for both languages.

following PPEs need to be satisfied:

$$(((T_1, T_2), (T_1', T_2')), (\hat{r}, \hat{t})) \in R_{\in} \iff e(g^{\xi}, \hat{r}) \cdot e(g^{-1}, \hat{t}) = 1_{\mathbb{G}_T} \ \wedge$$
$$\hat{r} \neq 1_{\mathbb{G}_2} \ \wedge \ \hat{t} \neq 1_{\mathbb{G}_2} \ \wedge \ e(T_2 \cdot T_2'^{-1}, \hat{r}) \cdot e(T_1^{-1} \cdot T_1', \hat{t}) = 1_{\mathbb{G}_T}. \quad (1)$$

By the soundness of the underlying $\mathcal{PKEQ}$ scheme, the PPEs above deliver the desired soundness properties for membership in $R_{\in}$. For membership in $R_{\notin}$, we have to exchange the last literal in the conjunction of the PPEs above by $e(T_2 \cdot T_2'^{-1}, \hat{r}) \cdot e(T_1^{-1} \cdot T_1', \hat{t}) \neq 1_{\mathbb{G}_T}$. It is important to note that an inequality (as in the second part of the conjunction) cannot be proven using GS.

Instantiation of $\Pi_{\in}$. We use the GS-based commit-and-prove scheme from [EG14]. Thereby, the advantage is that it is possible to reach composable zero-knowledge even when reusing commitments in proofs for different statements. Consequently, we can include commitments to $\hat{r}$ and $\hat{t}$ in the CRS and we can reuse these commitments to prove the satisfiability of the following PPE

$$\prod_{i=1}^{2} e(A_i, \hat{Y}_i) = e(T_2 \cdot T_2'^{-1}, \underline{\hat{r}}) \cdot e(T_1^{-1} \cdot T_1', \underline{\hat{t}}) = 1_{\mathbb{G}_T},$$

where the prover is given access to the openings of the commitments and the underlined values are not revealed to the verifier. The fact that the commitments are already contained in the CRS forces the prover to use commitments to the actual values which are consistent with the public key (instead of plugging in $\hat{r} = 1_{\mathbb{G}_2}, \hat{t} = 1_{\mathbb{G}_2}$ as the trivial solution).[2] The corresponding proof is very simple and can be communicated with two group elements in $\mathbb{G}_1$.[3] Since our instantiation is a straightforward application of the GS-based commit-and-prove scheme, we obtain the following lemma:

Lemma 3. *$\Pi_{\in}$ provides perfect completeness, perfect soundness and—because of the form of the PPE—composable zero-knowledge.*[4]

Instantiation of $\Pi_{\notin}$. To construct a proof for plaintext inequality statements, we build upon a recent technique by Blazy et al. [BCV15]. They proposed a generic way to (non-interactively) prove non-membership claims with respect to a language in zero-knowledge and provide multiple instantiations of their framework based on combinations of SPHFs and GS proofs. Informally, their

[2] For the simulation we may still use $\hat{r} = 1_{\mathbb{G}_2}, \hat{t} = 1_{\mathbb{G}_2}$.

[3] This is due to the fact that—for equations of this type—one can omit the $\boldsymbol{\pi}$ part of a GS proof $\pi = (\boldsymbol{\pi}, \boldsymbol{\theta})$ and only needs to send $\boldsymbol{\theta}$. In addition, due to the nature of the used projection map, the first components of the $\mathbb{G}_1^2$ elements in $\boldsymbol{\theta} \in \mathbb{G}_1^2 \times \mathbb{G}_1^2$ are $1_{\mathbb{G}_1}$, meaning that the proof only consists of two elements in $\mathbb{G}_1$.

[4] We note that, due to using the commit-and-prove approach from [EG14], we also use their composable zero-knowledge notion for commit-and-prove schemes. This notion can be seen as a generalization of standard composable zero-knowledge.

generic technique for proving non-membership works as follows. They use a non-interactive proof system Π_1 that allows to prove possession of a witness demonstrating the membership of some statement in some language, where the respective proof fails. Then, they use a non-interactive proof system Π_2 that allows to prove that $\Pi_1.\mathsf{Proof}$ has been computed honestly. This way, it is possible to express non-membership statements by producing a proof such that $\Pi_1.\mathsf{Verify}$ returns 0 and proving that the proof itself was honestly computed (since otherwise such a faulty proof would be trivially computable).

We will build our instantiation upon a SPHF for Π_1 (where we can use the SPHF framework from [BBC+13a], which allows to prove the required statements) and GS proofs for Π_2. However, in contrast to how this technique is used in [BCV15], in our setting the verifier does not know the randomness of the commitments. This imposes an additional technicality to be discussed below. In particular, we additionally compute the hash value H using $\mathsf{ProjHash}$ on the prover side and prove that H was honestly computed using an additional non-interactive zero-knowledge proof system Π_3 (which we instantiate with GS proofs). In Scheme 2, we present our non-interactive proof system for membership in a language $L_{R_{\notin}}$ that contains all tuples $(T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}})$, where the trapdoor committed to in $\boldsymbol{\mathcal{C}}_{\mathsf{tk}}$ allows to demonstrate plaintext inequality. For simplicity, crs is for Π_1 and Π_3.

$\mathbf{P} : L_{R_{\notin}}, (T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}) \in L_{R_{\notin}}, \boldsymbol{R}_{\mathsf{tk}}, \mathsf{crs}$		$\mathbf{V} : L_{R_{\notin}}, (T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}), \mathsf{crs}$
$hk \leftarrow \mathsf{HashKG}(L_R)$		
$hp \leftarrow \mathsf{ProjKG}(hk, L_R, (T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}))$		
$H' \leftarrow \mathsf{Hash}(hk, L_R, (T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}))$		
$\phi \leftarrow \Pi_2.\mathsf{Proof}((H' \wedge hp), hk)$		
$H \leftarrow \mathsf{ProjHash}(hp, L_R, (T, T', \mathsf{pk}, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}), \boldsymbol{R}_{\mathsf{tk}})$		
$\psi \leftarrow \Pi_3.\mathsf{Proof}((hp, H, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}), (\boldsymbol{R}_{\mathsf{tk}}))$	$\xrightarrow{hp, \phi, \psi, H, H'}$	$\Pi_2.\mathsf{Verify}(\phi) \ \wedge$
		$\Pi_3.\mathsf{Verify}(\psi) \stackrel{?}{=} 1 \ \wedge \ H \stackrel{?}{\neq} H'$

Scheme 2: NIPEI Proof System. $\mathbf{P}$ . . . Prover, $\mathbf{V}$ . . . Verifier.

A nice thing to note (which will allow us to improve the efficiency of $\Pi_{\notin}$) is that we do not need to simulate the proof ϕ. We will only require the proof to completely hide hk, i.e., to be witness indistinguishable.

Likewise to $\Pi_{\in}$, we can include the commitments $\boldsymbol{\mathcal{C}}_{\mathsf{tk}}$ to tk in the CRS and use these commitments in the SPHF. Accordingly, the corresponding PPE simplifies to $e(T_2 \cdot T_2'^{-1}, \underline{\hat{r}}) \cdot e(T_1^{-1} \cdot T_1', \underline{\hat{t}}) \neq 1_{\mathbb{G}_T}$. We additionally include commitments $\boldsymbol{\mathcal{C}}_{\boldsymbol{R}}$ to the randomness $\boldsymbol{R}_{\mathsf{tk}}$ used to compute $\boldsymbol{\mathcal{C}}_{\mathsf{tk}}$ in the CRS. Then we can use these commitments together with the GS-based commit-and-prove scheme from [EG14] to prove the honest computation of the projective hash value more efficiently. Likewise to the other commitments in the CRS, this ensures that the prover uses the correct values (while also ensuring the simulatability).

Since the instantiation of Π_1, Π_2, and Π_3 with the required properties is quite involved, we provide a detailed description in Appendix C and Appendix D, respectively. Finally, for Scheme 2 we can show the following:

Theorem 1. *If Π_1 is correct and the verifier cannot distinguish a failing proof (i.e., H) from random, Π_2 is complete, sound and witness indistinguishable, Π_3 is complete, sound and zero-knowledge, then $\Pi_{\notin}$ is also complete, sound and zero-knowledge.*

We prove Theorem 1 in Appendix D.3. By combining Lemma 2, Lemma 3, and Theorem 1 we straightforwardly derive the following corollary for our instantiation of the proof system $\Pi = (\Pi_{\in}, \Pi_{\notin})$.

Corollary 1. *The NIPEI proof system Π obtained by combining the above instantiations of $\Pi_{\in}$ and $\Pi_{\notin}$ is secure, i.e., complete, sound, and zero-knowledge.*

Instantiations with Other Encryption Schemes. For simplicity, we have presented an instantiation in the SXDH setting using ElGamal, but it is straightforward to adapt to Cramer-Shoup [CS98] or twin-ElGamal [DP06]. Furthermore, it is easy to adapt it to the DLIN setting and the corresponding linear encryption schemes.

4 GSSs with Verifiable Controllable Linkability

Subsequently, we propose a model for group signatures that considers verifiable controllable linkability and builds upon the model of Hwang et al. [HLhC+11, HLC+13, HCCN15] who formalized controllable linkability. Moreover, we consider the extension to the BSZ [BSZ05] model of Sakai et al. [SSE+12], i.e., opening soundness. The model involves three authorities: an issuing authority possessing the master issuing key (mik), an opening authority possessing the master opening key (mok), and a linking authority possessing the master linking key (mlk).

4.1 Model for GSSs with Verifiable Controllable Linkability

We now define GSSs with verifiable controllable linkability (VCL-GS).

Definition 4. *A VCL-GS is a tuple of efficient algorithms $\mathcal{GS} = (\mathsf{GkGen}, \mathsf{UkGen}, \mathsf{Join}, \mathsf{Issue}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge}, \mathsf{Link}, \mathsf{Judge}_{\mathsf{Link}})$, which are defined as follows.*

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

$\mathsf{UkGen}(1^\kappa)$: On input a security parameter κ, this algorithm generates a user key pair (usk_i, 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 of the group public key gpk, and 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 of 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 of the group public key gpk, a message M, and a signature σ, this algorithm verifies whether σ is valid with respect to M and gpk. If so, it outputs 1 and 0 otherwise.

$\mathsf{Open}(\mathsf{gpk}, \mathbf{reg}, M, \sigma, \mathsf{mok})$: On input of the group public key gpk, the registration table $\mathbf{reg}$, a message M, a valid signature σ, 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. If no group member produced σ, $\perp$ is returned.

$\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i, \mathsf{upk}_i, \tau)$: On input of the group public key gpk, a message M, a valid signature σ, the claimed signer i, the public key upk_i as well as a proof τ, this algorithm returns 1 if τ is a valid proof that i produced σ and 0 otherwise.

$\mathsf{Link}(\mathsf{gpk}, M, \sigma, M', \sigma', \mathsf{mlk})$: On input of the group public key gpk, a message M, a corresponding valid signature σ, a message M', a corresponding valid signature σ' and the master linking key mlk, this algorithm determines whether σ and σ' have been produced by the same or different signers and returns the linking decision $b \in \{1, 0\}$ as well as a publicly verifiable proof ρ attesting the validity of this decision.

$\mathsf{Judge}_{\mathsf{Link}}(\mathsf{gpk}, M, \sigma, M', \sigma', b, \rho)$: On input of the group public key gpk, a message M, a corresponding valid signature σ, a message M', a corresponding valid signature σ', a linking decision b as well as the corresponding linking proof ρ, this algorithm returns 1 if ρ is a valid proof for b with respect to σ and σ' and 0 otherwise.

Now we present the security properties for group signature schemes with verifiable controllable linkability. They are adopted from the model of Hwang et al. [HLhC+11, HLC+13, HCCN15] for GSSs with controllable linkability, which builds upon the BSZ [BSZ05] model.[5] In addition to the properties correctness, anonymity, non-frameability, and traceability defined in the BSZ model, Hwang et al. [HLhC+11, HLC+13, HCCN15] introduced properties to cover controllable linkability, namely LO-linkability (link-only linkability), JP-unforgeability (judge-proof unforgeability), and E-linkability (enforced linkability). Additionally, we integrate the proposal of Sakai et al. [SSE+12] who introduced the additional property of (weak) opening soundness as an optional property.[6] We briefly sketch them below and present formal definitions in Appendix E.

– **Anonymity:** Signers remain anonymous for all entities except for the opening authority.

[5] Actually, it uses a weaker anonymity notion similar to CPA-full anonymity [BBS04], where the challenge oracle can only be called once.

[6] We emphasize that this property is optional as there are no known GSSs with controllable linkability that have been shown to provide this property.

– **Traceability:** All valid signatures open correctly and allow to compute a valid opening proof.
– **Non-Frameability:** No entity is able to produce a valid opening proof that falsely accuses an honest user as the signer.
– **JP-Unforgeability:** The linking key is not useful to generate valid opening proofs.
– **LO-Linkability:** The linking key is only useful to link signatures, but not to open signatures.
– **E-Linkability:** Colluding users, linkers, and openers are not able to generate two message-signature pairs yielding contradicting opening and linking decisions.
– **Opening Soundness:** Colluding issuers, users, linkers, and openers are not able to produce two different (contradicting) opening proofs, even when allowed to corrupt users and/or the opener.[7]

In addition to the above, in the vein of Sakai et al., we introduce the additional notion of *linking soundness*. We only consider a strong variant, where the adversary has access to all keys. Informally, linking soundness targets contradicting linking proofs, where the signatures as well as the proofs may be maliciously generated, yet accepted by GVf and $\mathsf{Judge}_{\mathsf{Link}}$, respectively. In contrast, E-linkability targets contradicting results of Open and Link for maliciously generated signatures, where Open, Judge, and Link are honestly computed. Subsequently, we present a definition of linking soundness.

Definition 5 (Linking Soundness). *A group signature scheme $\mathcal{GS}$ with verifiable controllable linkability is said to provide linking soundness if for any adversary $\mathcal{A}$ and any $\kappa \in \mathbb{N}$, $\mathsf{Pr}[\mathsf{Exp}^{\mathsf{ls}}_{\mathcal{GS},\mathcal{A}}(\kappa) = 1] \leq \epsilon(\kappa)$.*

The experiment $\mathsf{Exp}^{\mathsf{ls}}_{\mathcal{GS},\mathcal{A}}$ is formally defined in Figure 1 of Appendix E.

4.2 Verifiable Controllable Linkability

Recall that in group signatures with controllable linkability the LA runs the Com algorithm of a $\mathcal{PKEQ}$ scheme to decide whether two ciphertexts contain the same unknown plaintext. Publishing the required trapdoor key tk would allow *any* party to link *any* two group signatures, which is clearly not desired. However, by means of our proposed NIPEI proof system we are able to allow the LA to prove whether or not any two signatures stem from the same signer without being able to identify the signer(s) and still only requiring tk.

Subsequently, we show how our generic construction for NIPEI proofs can be used to realize verifiable controllable linkability for group signatures following the SEP paradigm. Thereby, we assume that the used $\mathcal{PKEQ}$ is defined for bilinear groups, such that it is possible to set up the $\mathcal{PKEQ}$ and the proof systems in a compatible way. To this end, we assume that the group public key gpk contains a bilinear group description BG. Then, the modified group key generation algorithm GkGen' looks as follows:

[7] Note that Sakai et al. [SSE+12] also introduced a weaker version of this property denoted as weak opening soundness.

$\mathsf{GkGen}'(1^\kappa)$: Run $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$ and obtain BG from gpk. Then, run $\mathsf{crs} \leftarrow \Pi.\mathsf{CRSGen}(\mathsf{BG})$, set $\mathsf{gpk}' \leftarrow (\mathsf{gpk}, \mathsf{crs})$ and return (gpk', mok, mik, mlk).

Furthermore, the algorithms Link and $\mathsf{Link}_{\mathsf{Judge}}$ operate as follows:

$\mathsf{Link}(\mathsf{gpk}, M, \sigma, M', \sigma', \mathsf{mlk})$: Extract the ciphertexts T and T' from σ and σ', respectively. Obtain BG, pk_e from gpk and tk from mlk. Compute $\rho \leftarrow \Pi.\mathsf{Prove}(\mathsf{BG}, \mathsf{crs}, (T, T', \mathsf{pk}_e), \mathsf{tk})$ and return the linking decision b and the corresponding proof ρ.

$\mathsf{Link}_{\mathsf{Judge}}(\mathsf{gpk}, M, \sigma, M', \sigma', b, \rho)$: Extract the ciphertexts T and T' from σ and σ'. Obtain BG, crs and pk_e from gpk. If $b = 1$ and ρ is a proof for language $L_{R_{\notin}}$ or vice versa, return $\perp$. Otherwise, return $\Pi.\mathsf{Verify}(\mathsf{BG}, \mathsf{crs}, (T, T', \mathsf{pk}_e), \rho)$.

Security Analysis. We investigate to which extent the extension of a group signature scheme with controllable linkability (i.e., the constructions in [HLhC+11, HLC+13, HCCN15] and the generic conversion from [SSU14]) to one with verifiable controllable linkability requires to re-evaluate the original security properties. Note that the proof of the subsequent theorem is quite independent of the concrete definition of anonymity and works for group signature schemes providing the weaker anonymity notion by Hwang et al., but also with stronger notions such as CPA-full or CCA2-full anonymity (cf. the discussion in Appendix E).

Theorem 2. *Let* $\mathcal{GS} = (\mathsf{GkGen}, \mathsf{UkGen}, \mathsf{Join}, \mathsf{Issue}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge}, \mathsf{Link})$ *be a secure group signature scheme with controllable linkability with or without (weak) opening soundness, let* Π *be a secure NIPEI proof system, and let* $\mathcal{PKEQ} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Aut}, \mathsf{Com})$ *be the used* AoN-PKeet* *scheme, where* $\mathcal{PKEQ}$ *is compatible with* Π*. Then,* $\mathcal{GS}' = (\mathsf{GkGen}, \mathsf{UkGen}, \mathsf{Join}, \mathsf{Issue}, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Open}, \mathsf{Judge}, \mathsf{Link}, \mathsf{Judge}_{\mathsf{Link}})$ *is a secure group signature scheme with verifiable controllable linkability with or without (weak) opening soundness.*

We prove Theorem 2 in Appendix E.1.

Instantiating $\Pi_{\notin}$ for Group Signatures with Σ-Proofs. Many existing GSSs following the SEP paradigm are instantiated using the RO heuristic. Now, if one already relies on the ROM for the GSS, it might be an alternative to instantiate parts of $\Pi_{\notin}$ (i.e., Π_2 and Π_3) using a non-interactive Σ protocol obtained via the Fiat-Shamir transform, which is specifically crafted for the application with verifiable controllable linkability and the used SPHF instantiation. In Appendix D.4, we illustrate such an instantiation of $\Pi_{\notin}$.

References

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In *CRYPTO*, 2000.

[ACP09] Michel Abdalla, Céline Chevalier, and David Pointcheval. Smooth Projective Hashing for Conditionally Extractable Commitments. In *CRYPTO*, 2009.

[BBC+13a] Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. Efficient UC-Secure Authenticated Key-Exchange for Algebraic Languages. In *PKC*, 2013.

[BBC+13b] Fabrice Benhamouda, Olivier Blazy, Céline Chevalier, David Pointcheval, and Damien Vergnaud. New Techniques for SPHFs and Efficient One-Round PAKE Protocols. In *CRYPTO*, 2013.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short Group Signatures. In *CRYPTO*, 2004.

[BCC04] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct Anonymous Attestation. In *ACM CCS*. ACM, 2004.

[BCV15] Olivier Blazy, Céline Chevalier, and Damien Vergnaud. Non-Interactive Zero-Knowledge Proofs of Non-Membership. In *CT-RSA*, 2015.

[BFG+13] David Bernhard, Georg Fuchsbauer, Essam Ghadafi, Nigel P. Smart, and Bogdan Warinschi. Anonymous Attestation with User-Controlled Linkability. *Int. J. Inf. Sec.*, 12(3), 2013.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. In *CT-RSA*, 2005.

[CEJ+07] Seung Geol Choi, Ariel Elbaz, Ari Juels, Tal Malkin, and Moti Yung. Two-Party Computing with Encrypted Data. In *ASIACRYPT*, 2007.

[Cho09] Sherman S. M. Chow. Real Traceable Signatures. In *SAC*, 2009.

[CS97] Jan Camenisch and Markus Stadler. Efficient Group Signature Schemes for Large Groups. In *CRYPTO*, 1997.

[CS98] Ronald Cramer and Victor Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *CRYPTO*, 1998.

[CS02] Ronald Cramer and Victor Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In *EUROCRYPT*, 2002.

[CvH91] David Chaum and Eugène van Heyst. Group Signatures. In *EUROCRYPT*, 1991.

[DP06] Cécile Delerablée and David Pointcheval. Dynamic Fully Anonymous Short Group Signatures. In *VIETCRYPT*, 2006.

[EG14] Alex Escala and Jens Groth. Fine-Tuning Groth-Sahai Proofs. In *PKC*, 2014.

[EH14] Keita Emura and Takuya Hayashi. Road-to-Vehicle Communications with Time-Dependent Anonymity: A Light Weight Construction and its Experimental Results. Cryptology ePrint Archive, Report 2014/926, 2014.

[GL03] Rosario Gennaro and Yehuda Lindell. A Framework for Password-Based Authenticated Key Exchange. In *EUROCRYPT*, 2003.

[GS08] Jens Groth and Amit Sahai. Efficient Non-interactive Proof Systems for Bilinear Groups. In *EUROCRYPT*, 2008.

[HCCN15] Jung Yeon Hwang, Liqun Chen, Hyun Sook Cho, and DaeHun Nyang. Short Dynamic Group Signature Scheme Supporting Controllable Linkability. *IEEE Transactions on Information Forensics and Security*, 10(6), 2015.

[HLC+13] Jung Yeon Hwang, Sokjoon Lee, Byung-Ho Chung, Hyun Sook Cho, and DaeHun Nyang. Group Signatures with Controllable Linkability for Dynamic Membership. *Inf. Sci.*, 222, 2013.

[HLhC+11] Jung Yeon Hwang, Sokjoon Lee, Byung ho Chung, Hyun Sook Cho, and DaeHun Nyang. Short Group Signatures with Controllable Linkability. In *LightSec*. IEEE, 2011.

[IEH+15] Ai Ishida, Keita Emura, Goichiro Hanaoka, Yusuke Sakai, and Keisuke Tanaka. Group Signature with Deniability: How to Disavow a Signature. Cryptology ePrint Archive, Report 2015/043, 2015.

[JJ00] Markus Jakobsson and Ari Juels. Mix and Match: Secure Function Evaluation via Ciphertexts. In *ASIACRYPT*, 2000.

[KTY04] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable Signatures. In *EUROCRYPT*, 2004.

[MCVH12] Lukas Malina, Jordi Castellà-Roca, Arnau Vives-Guasch, and Jan Hajny. Short-Term Linkable Group Signatures with Categorized Batch Verification. In *FPS*, 2012.

[NFHF09] Toru Nakanishi, Hiroki Fujii, Yuta Hira, and Nobuo Funabiki. Revocable Group Signature Schemes with Constant Costs for Signing and Verifying. In *PKC*, 2009.

[NFW99] Toru Nakanishi, Toru Fujiwara, and Hajime Watanabe. A Linkable Group Signature and Its Application to Secret Voting. *Trans. of IPSJ*, 40(7), 1999.

[NY90] Moni Naor and Moti Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *STOC' 90*. ACM, 1990.

[PRST08] David C. Parkes, Michael O. Rabin, Stuart M. Shieber, and Christopher Thorpe. Practical Secrecy-Preserving, Verifiably Correct and Trustworthy Auctions. *Electronic Commerce Research and Applications*, 7(3), 2008.

[SEH+12] Yusuke Sakai, Keita Emura, Goichiro Hanaoka, Yutaka Kawai, Takahiro Matsuda, and Kazumasa Omote. Group Signatures with Message-Dependent Opening. In *Pairing*, 2012.

[SSE+12] Yusuke Sakai, Jacob C. N. Schuldt, Keita Emura, Goichiro Hanaoka, and Kazuo Ohta. On the Security of Dynamic Group Signatures: Preventing Signature Hijacking. In *PKC*, 2012.

[SSU14] Daniel Slamanig, Raphael Spreitzer, and Thomas Unterluggauer. Adding Controllable Linkability to Pairing-Based Group Signatures for Free. In *ISC*, 2014.

[Tan12a] Qiang Tang. Public Key Encryption Schemes Supporting Equality Test with Authorisation of Different Granularity. *IJACT*, 2(4), 2012.

[Tan12b] Qiang Tang. Public Key Encryption Supporting Plaintext Equality Test and User-Specified Authorization. *Security and Communication Networks*, 5(12), 2012.

[Wei05] Victor K. Wei. Tracing-by-Linking Group Signatures. In *ISC*, 2005.

A Computational Hardness Assumptions

Decisional Diffie-Hellman Assumption (DDH). Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p, such that $\log_2 p = \kappa$. Then, for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:

$$\Pr\left[\begin{matrix} b \xleftarrow{R} \{0,1\}, r, s, t \xleftarrow{R} \mathbb{Z}_p^*, \\ b^* \leftarrow \mathcal{A}(g, g^r, g^s, g^{b\cdot(rs)+(1-b)\cdot t}) \end{matrix} \;:\; b = b^*\right] \leq \frac{1}{2} + \epsilon(\kappa).$$

Decision Linear Assumption (DLIN). Let $\mathbb{G} = \langle u \rangle = \langle v \rangle = \langle h \rangle$ be a group of prime order p, such that $\log_2 p = \kappa$. Then, for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:

$$\Pr\left[\begin{array}{l} b \xleftarrow{R} \{0,1\}, r, s, t \xleftarrow{R} \mathbb{Z}_p, \\ b^* \leftarrow \mathcal{A}(u, v, h, u^r, v^s, h^{b\cdot(r+s)+(1-b)\cdot t}) \end{array} \;:\; b = b^*\right] \leq \frac{1}{2} + \epsilon(\kappa).$$

Computational co-Diffie-Hellman Assumption (co-CDH). Let $\mathbb{G}_1 = \langle g \rangle$, and $\mathbb{G}_2 = \langle \hat{g} \rangle$ be two distinct groups of prime order p, such that $\log_2 p = \kappa$. Then, for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:

$$\Pr\left[r \xleftarrow{R} \mathbb{Z}_p, \hat{h} \leftarrow \mathcal{A}(g, g^r, \hat{g}) \;:\; \hat{h} = \hat{g}^r\right] \leq \epsilon(\kappa).$$

External Diffie-Hellman Assumption (XDH). Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three cyclic groups of prime order p and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a pairing. Then, the XDH assumption states that the DDH assumption holds in $\mathbb{G}_1$.

Symmetric External Diffie-Hellman Assumption (SXDH). Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three cyclic groups of prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ a pairing. Then, the SXDH assumption states that the DDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$.

B Security of Non-Interactive Proof Systems

In the following, we state the security properties required for our construction (adapted from [GS08]).

Definition 6 (Perfect Completeness). *A non-interactive proof system is perfectly complete, if for all adversaries $\mathcal{A}$ it holds that*

$$\Pr\left[\begin{array}{l} \mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \\ \mathsf{crs} \leftarrow \mathsf{CRSGen}(\mathsf{BG}), \\ (x, w) \leftarrow \mathcal{A}(\mathsf{BG}, \mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{BG}, \mathsf{crs}, x, w) \end{array} \;:\; \begin{array}{r} \mathsf{Verify}(\mathsf{BG}, \mathsf{crs}, x, \pi) = 1 \\ \wedge\ (\mathsf{BG}, x, w) \in R \end{array}\right] = 1.$$

Definition 7 (Perfect Soundness). *A non-interactive proof system is perfectly sound, if for all adversaries $\mathcal{A}$ it holds that*

$$\Pr\left[\begin{array}{l} \mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \\ \mathsf{crs} \leftarrow \mathsf{CRSGen}(\mathsf{BG}), \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{BG}, \mathsf{crs}) \end{array} \;:\; \begin{array}{r} \mathsf{Verify}(\mathsf{BG}, \mathsf{crs}, x, \pi) = 1 \\ \wedge\ x \notin L_R \end{array}\right] = 0.$$

Definition 8 (Witness Indistinguishability). *A non-interactive proof system is composably witness indistinguishable, if there exists a simulator S such that for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:*

$$\left|\begin{array}{l} \Pr\left[\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \mathsf{crs} \leftarrow \mathsf{CRSGen}(\mathsf{BG}) : \mathcal{A}(\mathsf{BG}, \mathsf{crs}) = 1\right] - \\ \Pr\left[\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \mathsf{crs} \leftarrow S(\mathsf{BG}) : \mathcal{A}(\mathsf{BG}, \mathsf{crs}) = 1\right] \end{array}\right| \leq \epsilon(\kappa),$$

and

$$\Pr\left[\begin{array}{l} b \xleftarrow{R} \{0,1\}, \mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \\ \mathsf{crs} \leftarrow S(\mathsf{BG}), (x, w_0, w_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{BG}, \mathsf{crs}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{BG}, \mathsf{crs}, x, w_b), b^* \leftarrow \mathcal{A}(\pi, \mathsf{st}) \end{array} : \begin{array}{r} b = b^* \wedge \\ (\mathsf{BG}, x, w_0) \in R \wedge \\ (\mathsf{BG}, x, w_1) \in R \end{array}\right] = \frac{1}{2}.$$

Definition 9 (Zero-Knowledge). *A non-interactive proof system is composably zero-knowledge, if there exist two simulators S_1 and S_2 such that for all PPT adversaries $\mathcal{A}$ there exists a negligible function $\epsilon(\cdot)$ such that:*

$$\left|\begin{array}{l} \Pr\left[\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), \mathsf{crs} \leftarrow \mathsf{CRSGen}(\mathsf{BG}) : \mathcal{A}(\mathsf{BG}, \mathsf{crs}) = 1\right] - \\ \Pr\left[\mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), (\mathsf{crs}, \mathsf{T}) \leftarrow S_1(\mathsf{BG}) : \mathcal{A}(\mathsf{BG}, \mathsf{crs}) = 1\right] \end{array}\right| \leq \epsilon(\kappa),$$

and

$$\Pr\left[\begin{array}{l} \mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), (\mathsf{crs}, \mathsf{T}) \leftarrow S_1(\mathsf{BG}), \\ (x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{BG}, \mathsf{crs}, \mathsf{T}), \\ \pi \leftarrow \mathsf{Proof}(\mathsf{BG}, \mathsf{crs}, x, w) \end{array} : \mathcal{A}(\pi, \mathsf{st}) = 1\right] =$$

$$\Pr\left[\begin{array}{l} \mathsf{BG} \leftarrow \mathsf{BGGen}(1^\kappa), (\mathsf{crs}, \mathsf{T}) \leftarrow S_1(\mathsf{BG}), \\ (x, w, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{BG}, \mathsf{crs}, \mathsf{T}), \\ \pi \leftarrow S_2(\mathsf{BG}, \mathsf{crs}, x, \mathsf{T}) \end{array} : \mathcal{A}(\pi, \mathsf{st}) = 1\right].$$

Here, T denotes a simulation trapdoor.

C Security and Instantiation of SPHFs

Security Properties. The correctness requires that $\mathsf{Hash}(hk, L, W) = \mathsf{ProjHash}(hp, L, W, w)$ for all $W \in L$ and their corresponding witnesses w. Smoothness requires that if $W \notin L$, the following distributions are statistically indistinguishable:

$$\begin{aligned} &\{(L, \mathsf{pp}, W, hp, H') \mid \mathsf{param} \leftarrow \mathsf{Setup}(1^\kappa), hk \leftarrow \mathsf{HashKG}(L), \\ &\qquad hp \leftarrow \mathsf{ProjKG}(hk, L, W), H' \leftarrow \mathsf{Hash}(hk, L, W)\} \approx \\ &\{(L, \mathsf{pp}, W, hp, H') \mid \mathsf{param} \leftarrow \mathsf{Setup}(1^\kappa), hk \leftarrow \mathsf{HashKG}(L), \\ &\qquad hp \leftarrow \mathsf{ProjKG}(hk, L, W), H' \xleftarrow{R} Dom(\mathsf{Hash})\} \end{aligned}$$

The pseudo-randomness requires that if $W \in L$, then without a witness of membership the distributions considered in smoothness remain computationally indistinguishable. We call a SPHF secure if it satisfies all the above properties.

Languages that can be used with SPHFs have recently been extended to every kind of pairing product equations over graded rings [BBC+13a]. These constructions build upon SPHFs on linear Cramer-Shoup ciphertexts [GL03, BBC+13a], but can easily be adapted to the SXDH setting.

Instantiation of the SPHF. For efficiency reasons, we use ElGamal commitments together with the SPHF on ElGamal ciphertexts as proposed by Gennaro and Lindell [GL03]. In Scheme 3, we recall the aforementioned SPHF for

the group $\mathbb{G}_2$ in an SXDH setting, where the language L contains all triples (pk, C_M, M) of ElGamal public keys pk, ciphertexts C_M with respect to pk and corresponding messages M, where membership in this language is witnessed by the used randomness r. Applying the techniques presented in [BBC+13b,

$\mathsf{Setup}(1^\kappa)$: On input of κ, this algorithm generates a bilinear group description $\mathsf{BG} \leftarrow (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, \hat{g})$ such that $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of common prime order p with $\log_2 p = \kappa$, e is a pairing, $\mathbb{G}_1 = \langle g \rangle$ and $\mathbb{G}_2 = \langle \hat{g} \rangle$. It returns $\mathsf{pp} \leftarrow \mathsf{BG}$.
$\mathsf{HashKG}(L)$: On input of pp and L, this algorithm returns $hk \leftarrow (\eta, \theta) \xleftarrow{R} \mathbb{Z}_p^2$.
$\mathsf{ProjKG}(hk, L, W)$: On input of pp, hk, L and some word $W = (\mathsf{pk}, C_M, M)$, where $C_M = (\hat{g}^r, M \cdot \mathsf{pk}^r)$ this algorithm computes and returns $hp \leftarrow \hat{g}^\eta \mathsf{pk}^\theta$.
$\mathsf{Hash}(hk, L, W)$: On input of pp, hk, L and $W = (\mathsf{pk}, C_M, M) \in \mathbb{G}_2 \times \mathbb{G}_2^2 \times \mathbb{G}_2$, where $C_M = (\hat{u}, \hat{e})$, this algorithm computes and returns $H' \leftarrow \hat{u}^\eta (\hat{e}/M)^\theta$.
$\mathsf{ProjHash}(hp, L, W, w)$: On input of pp, hp, L, W, $w = r$, this algorithm computes and returns $H \leftarrow hp^r$.

Scheme 3: SPHF on ElGamal Ciphertexts [GL03]

BBC+13a] to the SPHF in Scheme 3, allows to prove satisfiability of pairing product equations. In our setting, we have PPEs of the form

$$\prod_{i=1}^{2} (A_i, \underline{\hat{Y}_i}) = 1_{\mathbb{G}_T}, \tag{2}$$

where $\hat{Y}_i$ are not revealed to the verifier and are thus encrypted using ElGamal (further denoted as $C_i = (\hat{u}_i, \hat{e}_i) = (\hat{g}^{r_i}, \hat{Y}_i \cdot \mathsf{pk}^{r_i})$). Let the setup be as follows: $\theta \xleftarrow{R} \mathbb{Z}_p$ and for $i \in \{1, 2\}$, $\eta_i \xleftarrow{R} \mathbb{Z}_p$, $hk_i = (\eta_i, \theta) \in \mathbb{Z}_p^2$ as well as $hp_i = \hat{g}^{\eta_i} \mathsf{pk}^\theta$, where $hp = (hp_1, hp_2)$ is handed over to the prover. Then the SPHF is defined as follows

$$H' = \prod_{i=1}^{2} e(A_i, \hat{u}_i^{\eta_i} \hat{e}_i^\theta) = \prod_{i=1}^{2} (A_i, hp_i^{r_i}) = H.$$

Subsequently, we prove the following:

Lemma 4. *The SPHF described above is a secure SPHF for a language defined by Equation 2.*

Subsequently, we show correctness, smoothness, and pseudo-randomness.

Proof (Correctness). Let r_1 and r_2 be the randomness used to compute the encryptions of $\hat{Y}_1$ and $\hat{Y}_2$, respectively (which represents the witness). Then, the projective hash value using the projection key $hp_i = \hat{g}^{\eta_i} \mathsf{pk}^\theta$ for $i \in \{1, 2\}$ is computed as

$$H \leftarrow \prod_{i=1}^{2} (A_i, hp_i^{r_i}) = \prod_{i=1}^{2} e(A_i, (\hat{g}^{\eta_i} \mathsf{pk}^\theta)^{r_i})$$

Computing the hash value using the hashing key $hk_i = (\eta_i, \theta) \in \mathbb{Z}_p^2$ for $i \in \{1, 2\}$ yields:

$$H' \leftarrow \prod_{i=1}^{2} e(A_i, \hat{u}_i^{\eta_i} \hat{e}_i^{\theta}) = \prod_{i=1}^{2} e(A_i, \hat{g}^{\eta_i r_i} \cdot \hat{Y}_i^{\theta} \cdot \mathsf{pk}^{r_i \theta}) = \prod_{i=1}^{2} e(A_i, (\hat{g}^{\eta_i} \mathsf{pk}^{\theta})^{r_i}) \cdot e(A_i, \hat{Y}_i^{\theta}) \overset{(ii)}{=} H,$$

where for the last step (ii), we know that $\prod_{i=1}^{2} e(A_i, \hat{Y}_i) = 1_{\mathbb{G}_T}$ by definition. □

While the PPE above is tailored to our specific setting, this approach also works for arbitrary $i \in \mathbb{N}$. Subsequently, we prove smoothness (for the general case) by adapting the proof strategy from [BBC+13a, GL03]:

Proof (Smoothness). We can without loss of generality assume that one of the n ElGamal commitments encrypts a value such that the overall PPE is not fulfilled. In particular, this means that at least one commitment is of the form $(g^{r_i}, \hat{Y}_i \cdot \mathsf{pk}^{r_i'}) \in \mathbb{G}_2^2$, where $r_i \neq r_i'$. With $hp_i = \hat{g}^{\eta_i} \mathsf{pk}^{\theta}$, the hash value H' is defined to be:

$$H' \leftarrow \prod_{i=1}^{n} e(A_i, \hat{h}_i), \text{ where } \hat{h}_i = (\hat{g}^{r_i \eta_i} (\hat{Y}_i \cdot \mathsf{pk}^{r_i'})^{\theta}).$$

We can rewrite $\hat{h}_i$ as $\hat{h}_i = (\hat{g}^{r_i \eta_i} \mathsf{pk}^{r_i' \theta}) \cdot \hat{Y}_i^{\theta}$. Assuming that $\hat{Y}_i^{\theta}$ cancels out when plugging into the pairing product equation, it suffices to consider the following discrete logarithms:

$$\log_{\hat{g}} \hat{h}_i = r_i \eta_i + r_i' x_{\mathsf{pk}} \theta, \text{ and } \log_{\hat{g}} hp_i = \eta_i + x_{\mathsf{pk}} \theta.$$

As we know that $r_i \neq r_i'$ we have that these two values are linearly independent, which shows that H' looks perfectly random. □

Proof (Pseudo-Randomness). We know that smoothness holds. We prove pseudo-randomness by showing that a distinguisher between the distributions considered in smoothness and pseudo-randomness is a distinguisher for DDH. For simplicity we assume that $i = 1$. We obtain a DDH instance $(\hat{g}^r, \hat{g}^s, \hat{g}^t) \in \mathbb{G}_2^3$ and compute the ciphertext to $\hat{Y}_1$ as $(\hat{g}^r, \hat{Y}_1 \cdot \hat{g}^t)$, set $\mathsf{pk} \leftarrow \hat{g}^s$, choose $hk_1 = (\eta, \theta) \xleftarrow{R} \mathbb{Z}_p^2$ and set $hp_1 \leftarrow \hat{g}^{\eta} \mathsf{pk}^{\theta}$. Consequently, if we have a valid DDH instance, we have a distribution as in the pseudo-randomness game, whereas we have a distribution as in the smoothness game if the DDH instance is random. □

By a standard hybrid argument, this proof can be extended to arbitrary $i \in \mathbb{N}$.

D Instantiation of Non-Interactive Proof Systems for $\Pi_{\notin}$

For our instantiations of Π_2 and Π_3 we use the following observation to obtain the zero-knowledge property for $\Pi_{\notin}$: The value H' (computed using Hash) looks

perfectly random to the verifier, whereas the value H (computed using $\mathsf{ProjHash}$) looks computationally random to the verifier. Furthermore, the values H and H' are not fixed prior to the proof, meaning that the prover can choose them at the time when computing the proof to ensure simulatability.

Let us briefly recall the setup with $i \in \{1,2\}$: $hk_i = (\eta_i, \theta) \in_R \mathbb{Z}_p^2, hp_i = \hat{g}^{\eta_i} \mathsf{pk}^{\theta} \in \mathbb{G}_2, H' = \prod^i e(A_i, \hat{u}_i^{\eta_i} \hat{e}_i^{\theta}) = e(A_1, \hat{u}_1^{\eta_1} \hat{e}_1^{\theta}) \cdot e(A_2, \hat{u}_2^{\eta_2} \hat{e}_2^{\theta}) = e(A_1, \hat{h}_1) \cdot e(A_2, \hat{h}_2) \in \mathbb{G}_T, \mathcal{C}_i = (\hat{u}_i, \hat{e}_i) = (\hat{g}^{r_i}, \hat{Y}_i \cdot \mathsf{pk}^{r_i}) \in \mathbb{G}_2^2$, where $\hat{Y}_1 = \hat{r}$ and $\hat{Y}_2 = \hat{t}$, $H = \prod^i e(A_i, hp_i^{r_i})$.

Here, A_1, A_2, $hp = (hp_1, hp_2), \boldsymbol{\mathcal{C}}_{\mathsf{tk}} = (\mathcal{C}_1, \mathcal{C}_2)$ are available to the prover and the verifier and the verifier also learns H' and H during the proof. The values r_i, hk_i and $\hat{h} = (\hat{h}_1, \hat{h}_2)$ are only known to the prover.

D.1 Instantiation of Π_2

To prove the consistency of H' and hp with respect to hk using the GS commit-and-prove approach, we prove the validity of the following conjunction of PPEs, where the underlined values are not revealed to the verifier:

$$e(\underline{g}, hp_i) = e(\underline{g^{\eta_i}}, \hat{g}) \cdot e(\underline{g^{\theta}}, \mathsf{pk}) \ \wedge \ e(\underline{g}, \underline{\hat{h}_i}) = e(\underline{g^{\eta_i}}, \hat{u}_i) \cdot e(\underline{g^{\theta}}, \hat{e}_i) \\ \wedge \ H' = \prod^i e(A_i, \underline{\hat{h}_i}).$$

As we only require witness indistinguishability for Π_2, the GS framework allows to straightforwardly prove equations of the form above. We require the commitment to g for the first and the second literal to be included in the CRS to ensure that the prover uses the correct values while ensuring enough freedom for the simulator of our overall proof system $\Pi_{\notin}$.

D.2 Instantiation of Π_3

When proving the consistency of H and hp with respect to r_i using the GS commit-and-prove approach, one can observe that—as long as hk is unknown—the values hp_i constitute random values in $\mathbb{G}_2$ and the values $hp_i^{r_i}$ look random under DDH in $\mathbb{G}_2$. Thus, one only has to prove the satisfiability of

$$e(\underline{g^{r_i}}, hp_i) = e(\underline{g}, hp_i^{r_i}), \tag{3}$$

and the verifier can compute $H \leftarrow \prod^i e(A_i, hp_i^{r_i})$ on its own. The commitments to g and g^{r_i} are included in the CRS to ensure that the prover uses the correct values while the simulator can use commitments to $1_{\mathbb{G}_1}$.

Using standard Groth-Sahai techniques, this PPE admits efficient non-interactive zero-knowledge proofs.

We note that although the Groth-Sahai framework allows us to prove a statement like Equation (3) in a zero-knowledge way, we still need to argue that revealing the $hp_i^{r_i}$ does not reveal too much information in the global scheme (as we do subsequently in the proof of Theorem 1).

D.3 Proof of Theorem 1

Proof. While completeness follows from inspection, we subsequently prove soundness and zero-knowledge.

Soundness: We setup the CRSs of the GS proof systems Π_2 and Π_3 so that they provide perfect soundness. Under the CRS indistinguishability of GS proofs, the prover will only be able to distinguish this setup from the perfect witness indistinguishability/zero knowledge setup with negligible probability. Then the perfect soundness of Π_2 and Π_3 guarantees the honest computation of the hash values H and H', whereas the perfect correctness of the SPHF guarantees that $H \neq H'$ only holds if the proven statement is actually true. Thus, the overall proof system provides perfect soundness as long as the CRS indistinguishability holds. □

Zero-knowledge: We setup the CRSs of the GS proof systems Π_2 and Π_3 so that they provide perfect witness indistinguishability/zero knowledge. Under the CRS indistinguishability of GS proofs, the prover will only be able to distinguish this setup from the perfect soundness setup with negligible probability. Subsequently, we prove zero-knowledge using a sequence of games.

Game 0: The original zero-knowledge game.

Game 1: We pick $hp_i \xleftarrow{R} \mathbb{G}_2$. Furthermore, we use random values $\hat{h}_i \xleftarrow{R} \mathbb{G}_2$ inside $\underline{\hat{h}_i}$, defining a random H'. Finally, instead of the actual commitments to $\underline{g}$, $\underline{g^{\eta_i}}$, and $\underline{g^{\theta}}$ we use commitments to $1_{\mathbb{G}_1}$.

Transition Game 0 → Game 1: Since the witness indistinguishability of Π_2 is perfect, the adversary will not be able to detect this game change (H' already looks perfectly random under the smoothness of the SPHF; $H = H'$ will only occur with negligible probability).

Game 2: We simulate the GS proof ψ and, thereby, use commitments to $1_{\mathbb{G}_1}$ instead of the actual values $\underline{g}$ and $\underline{g^{r_i}}$.

Transition Game 1 → Game 2: Since the zero-knowledge property of Π_3 is perfect, the adversary will not be able to detect this game change.

Game 3: We pick $hp_i^{r_i} \xleftarrow{R} \mathbb{G}_2$.

Transition Game 2 → Game 3: Besides $hp_i^{r_i} \xleftarrow{R} \mathbb{G}_2$, the only components depending on the values r_i are contained in $\boldsymbol{\mathcal{C}}_{\mathsf{tk}} = (\mathcal{C}_1, \mathcal{C}_2) = ((\hat{g}^{r_1}, \hat{Y}_1 \cdot \mathsf{pk}^{r_1}), (\hat{g}^{r_2}, \hat{Y}_2 \cdot \mathsf{pk}^{r_2}))$. To see that Game 2 and Game 3 are indistinguishable, assume that the simulator obtains a DDH instance $(\hat{g}, \hat{g}^{s_1}, \hat{g}^{t_1}, \hat{g}^{\mathsf{r}_1})$, and—by the random self reducibility of DDH—generates a second instance $(\hat{g}, \hat{g}^{s_2}, \hat{g}^{t_2}, \hat{g}^{\mathsf{r}_2})$. For $i \in \{1, 2\}$, it sets $(\hat{g}, \hat{g}^{r_i}, hp_i, hp_i^{r_i}) \leftarrow (\hat{g}, \hat{g}^{s_i}, \hat{g}^{t_i}, \hat{g}^{\mathsf{r}_i})$ and computes $\hat{Y}_i \cdot \mathsf{pk}^{r_i}$ by using the secret key sk corresponding to pk, i.e., as $\hat{Y}_i \cdot (\hat{g}^{r_i})^{\mathsf{sk}}$. Then, if the DDH instance is valid we have a distribution as in Game 2, whereas we have a distribution as in Game 3 if the DDH instance is random. Again $H = H'$ only occurs with negligible probability.

Game 4: We replace the commitments to tk, i.e., $\boldsymbol{\mathcal{C}}_{\mathsf{tk}}$, in the CRS by commitments to random values.

Transition Game 3 → Game 4: By the hiding property of the commitments,

this game change will remain indistinguishable for the adversary (again, $H = H'$ only occurs with negligible probability).

The last game represents a game that is simulated without knowledge of the trapdoor tk, whereas the first game represents the real game. Both games are indistinguishable, which proves the zero-knowledge property. □

D.4 Instantiation of Π_2 and Π_3 with Σ-Protocols

Let us recall where we require WI or NIZK proofs in $\Pi_{\notin}$: Namely, a WI proof ϕ to prove the consistency of the hash value, the projection key and the hash key as well as a NIZK proof ψ to prove the consistency of the hash value, the projection key and the commitments in the CRS. Furthermore, recall that the crs contains two commitments $\mathcal{C}_1 = (\hat{u}_1, \hat{e}_1) = (\hat{g}^{r_1}, \hat{r} \cdot \mathsf{pk}^{r_1})$, $\mathcal{C}_2 = (\hat{u}_2, \hat{e}_2) = (\hat{g}^{r_2}, \hat{t} \cdot \mathsf{pk}^{r_2})$, where the hashing/projection keys are chosen as $hk_i = (\eta_i, \theta)$ and $hp_i = \hat{g}^{\eta_i} \mathsf{pk}^{\theta}$, $i \in \{1, 2\}$. Since we can as well instantiate ϕ as a NIZK proof, we combine the predicates for ϕ and ψ straightforwardly using an AND composition and obtain the following relation, where $\hat{v}_i = hp_i^{r_i}$:

$$\mathsf{PoK}\Big\{(\eta_1, \eta_2, \theta, \rho_1, \rho_2) : H' = \prod_{i=1}^{2} e(A_i, \hat{u}_i^{\eta_i} \hat{e}_i^{\theta}) \ \wedge \ hp_1 = \hat{g}^{\eta_1} \mathsf{pk}^{\theta} \ \wedge \\ hp_2 = \hat{g}^{\eta_2} \mathsf{pk}^{\theta} \ \wedge \ \hat{v}_1 = hp_1^{\rho_1} \ \wedge \ \hat{v}_2 = hp_2^{\rho_2} \ \wedge \ \hat{u}_1 = \hat{g}^{\rho_1} \ \wedge \ \hat{u}_2 = \hat{g}^{\rho_2}\Big\},$$

and the verifier additionally checks whether $\prod^{i} e(A_i, \hat{v}_i) \neq H'$. The proof above can be instantiated by only revealing $H', A_i, \boldsymbol{\mathcal{C}}_{\mathsf{tk}}, hp_i, hp_i^{r_i}$ to the verifier.

E Security for VCL Group Signatures

In the following, **CU**, **HU**, **GSet** represent the set of corrupted users, honest users, and message-signature pairs corresponding to queries to the challenge oracle, respectively. Furthermore, **REG** is the list of transcripts generated by the join process. The oracles $\mathtt{SndToU}, \mathtt{AddU}, \mathtt{WReg}, \mathtt{RReg}, \mathtt{GSig}, \mathtt{USK}$, and $\mathtt{CrptU}$ represent a send to user oracle (by a corrupted issuer), an add user oracle, a write registration table oracle, a read registration table oracle, a group signing oracle, a user secret key oracle, and a corrupt user oracle. Furthermore, $\mathtt{Ch}_b$ represents the challenge oracle. We note that the model of Hwang et al. [HLhC+11, HLC+13, HCCN15] does only consider a weaker version of CPA-full-anonymity [BBS04], i.e., the adversary has no access to the $\mathtt{Open}$ oracle and the challenge oracle can only be called once. Without those restrictions, one can achieve the stronger notions of CPA-full and CCA2-full-anonymity (as defined in the BSZ model [BSZ05]) for this model.

Experiment $\mathsf{Exp}^{\text{anon-b}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$, $\mathbf{GSet} \leftarrow \emptyset$;
- $(i_0, i_1, M) \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{Link},\mathtt{WReg},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik})$;
- $\sigma_{i_b} \leftarrow \mathtt{Ch}_b(i_0, i_1, M)$;
- $b' \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{Link},\mathtt{WReg},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \sigma_{i_b})$, where $\mathcal{A}$ queried neither $\mathtt{Link}(\sigma_{i_b}, M, \cdot, \cdot)$ nor $\mathtt{Link}(\cdot, \cdot, \sigma_{i_b}, M)$;
- return b';

Experiment $\mathsf{Exp}^{\text{trace}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$;
- $(M, \sigma) \leftarrow \mathcal{A}^{\mathtt{SndToI},\mathtt{AddU},\mathtt{RReg},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mok}, \mathsf{mlk})$;
- If $\mathsf{GVf}(\mathsf{gpk}, M, \sigma) = 0$ then return 0;
- $(i, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathbf{REG}, M, \sigma, \mathsf{mok})$;
- If $i = 0$ or $\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i, \mathbf{upk}[i], \tau) = 0$ then return 1 else return 0;

Experiment $\mathsf{Exp}^{\text{nf}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$;
- $(M, \sigma, i) \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{AddU},\mathtt{RReg},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mok}, \mathsf{mlk})$;
- If $\mathsf{GVf}(\mathsf{gpk}, M, \sigma) = 0$ then return 0;
- If all of the following conditions hold, then return 1 and 0 otherwise
  - $i \in \mathbf{HU}$ and $\mathbf{usk}[i] \neq \emptyset$;
  - $(i, \tau) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathbf{REG}, M, \sigma, \mathsf{mok})$;
  - $\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i, \mathbf{upk}[i], \tau) = 1$;
  - $\mathcal{A}$ did not query $\mathtt{USK}(i)$ or $\mathtt{GSig}(i, M)$.

Experiment $\mathsf{Exp}^{\text{lo-link}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$, $\mathbf{GSet} \leftarrow \emptyset$;
- $(i_0, i_1, M) \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{AddU},\mathtt{GSig},\mathtt{Open},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mlk})$;
- $\sigma_{i_b} \leftarrow \mathtt{Ch}_b(i_0, i_1, M)$;
- $b' \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{AddU},\mathtt{GSig},\mathtt{Open},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, M, \sigma_{i_b})$;
- If all of the following conditions hold, then return 1 and 0 otherwise
  - $i_0, i_1 \in \mathbf{HU}$;
  - $\mathtt{GSig}(i_0, \cdot)$ and $\mathtt{GSig}(i_1, \cdot)$ and $\mathtt{Open}(M, \sigma)$ have not been queried;
  - $b' = b$

Experiment $\mathsf{Exp}^{\text{jp-uf}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$, $\mathbf{GSet} \leftarrow \emptyset$;
- $(i, M) \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{AddU},\mathtt{WReg},\mathtt{GSig},\mathtt{Open},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mlk})$;
- $\sigma \leftarrow \mathsf{GSig}(\mathsf{gpk}, i, M)$;
- $\tau \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{WReg},\mathtt{GSig},\mathtt{Open},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mlk}, i, M, \sigma)$;
- If all of the following conditions hold, then return 1 and 0 otherwise
  - $i \in \mathbf{HU}$ and $\mathbf{gsk}[i] \neq \emptyset$;
  - $\mathtt{Open}(M, \sigma)$ has not been queried;
  - $\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i, \mathbf{upk}[i], \tau) = 1$

Experiment $\mathsf{Exp}^{\text{os}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$;
- $(i_0, \tau_0, i_1, \tau_1, M, \sigma) \leftarrow \mathcal{A}^{\mathtt{WReg},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mok}, \mathsf{mlk})$;
- If $\mathsf{GVf}(\mathsf{gpk}, M, \sigma) = 0$ then return 0;
- If all of the following conditions hold, then return 1 and 0 otherwise
  - $i_0 \neq i_1$;
  - $\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i_0, \mathbf{upk}[i_0], \tau_0) = 1$;
  - $\mathsf{Judge}(\mathsf{gpk}, M, \sigma, i_1, \mathbf{upk}[i_1], \tau_1) = 1$;

Experiment $\mathsf{Exp}^{\text{e-link}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$, $\mathbf{GSet} \leftarrow \emptyset$;
- $(M_0, \sigma_0, M_1, \sigma_1) \leftarrow \mathcal{A}^{\mathtt{SndToU},\mathtt{AddU},\mathtt{RReg},\mathtt{GSig},\mathtt{USK},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mok}, \mathsf{mlk})$;
- If $\mathsf{GVf}(\mathsf{gpk}, M_0, \sigma_0) = 0$ or $\mathsf{GVf}(\mathsf{gpk}, M_1, \sigma_1) = 0$ then return 0;
- $(i_0, \tau_{i_0}) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathbf{REG}, M_0, \sigma_0, \mathsf{mok})$
- $(i_1, \tau_{i_1}) \leftarrow \mathsf{Open}(\mathsf{gpk}, \mathbf{REG}, M_1, \sigma_1, \mathsf{mok})$
- If $\mathsf{Judge}(\mathsf{gpk}, i_0, \mathbf{upk}[i_0], M_0, \sigma_0, \tau_{i_0}) = 0$ or $\mathsf{Judge}(\mathsf{gpk}, i_1, \mathbf{upk}[i_1], M_1, \sigma_1, \tau_{i_1}) = 0$ then return 0;
- $(b, \rho) \leftarrow \mathsf{Link}(\mathsf{gpk}, M_0, \sigma_0, M_1, \sigma_1, \mathsf{mlk})$
- If $\mathsf{Judge}_{\mathsf{Link}}(\mathsf{gpk}, M_0, \sigma_0, M_1, \sigma_1, b, \rho) = 0$ then return 0;
- If $i_0 \neq i_1$ and $b = 1$ (i.e., the two signatures link) then return 1;
- elif $i_0 = i_1$ and $b = 0$ (i.e., the two signatures do not link) then return 1;
- else return 0;

Experiment $\mathsf{Exp}^{\text{ls}}_{\mathcal{GS},\mathcal{A}}(\kappa)$:

- $(\mathsf{gpk}, \mathsf{mok}, \mathsf{mik}, \mathsf{mlk}) \leftarrow \mathsf{GkGen}(1^\kappa)$, $\mathbf{CU} \leftarrow \emptyset$, $\mathbf{HU} \leftarrow \emptyset$;
- $(s, M_0, \sigma_0, M_1, \sigma_1, b, \rho, b', \rho') \leftarrow \mathcal{A}^{\mathtt{WReg},\mathtt{CrptU}}(\mathsf{gpk}, \mathsf{mik}, \mathsf{mok}, \mathsf{mlk})$;
- If $\mathsf{GVf}(\mathsf{gpk}, M_0, \sigma_0) = 0$ or $\mathsf{GVf}(\mathsf{gpk}, M_1, \sigma_1) = 0$ then return 0;
- If $\mathsf{Judge}_{\mathsf{Link}}(\mathsf{gpk}, M_0, \sigma_0, M_1, \sigma_1, b, \rho) = 0$ then return 0;
- If $\mathsf{Judge}_{\mathsf{Link}}(\mathsf{gpk}, M_0, \sigma_0, M_1, \sigma_1, b', \rho') = 0$ then return 0;
- If $b \neq b'$, then return 1;
- else return 0;

Fig. 1. Experiments to define anonymity, traceability, non-frameability, link-only linkability, judge-proof unforgeability, enforced linkability (cf. Hwang et al. [HLhC+11, HLC+13, HCCN15]; enforced linkability is adapted to fit VCL-GS), opening soundness (cf. Sakai et al. [SSE+12]), and linking soundness.

E.1 Proof of Theorem 2

Proof. In all security games except the anonymity experiment, the adversary is in possession of the master linking key mlk. Thus, it is easy to see that the existing security notions with exception of anonymity are not influenced by the modifications. Now, what remains is to show that anonymity and that the new notion of linking soundness holds.

Anonymity. We show that anonymity holds using a sequence of games, where the event that the adversary wins Game i is denoted by S_i:

Game 0: The original anonymity game.

Game 1: As the original game, but the proofs obtained from the linking oracle are simulated[8] without using the trapdoor key tk.

Transition: Game 0 → Game 1: By the zero-knowledge property of Π, the adversary will only be able to distinguish Game 0 and Game 1 with negligible probability, i.e., $|\Pr[S_0] - \Pr[S_1]| \leq \epsilon_{\mathsf{zk}}(\kappa)$.

It is easy to see that the advantage of the adversary in Game 1 is the same as in the anonymity game without extensions. Observe that the signatures obtained from $\mathtt{Ch}_b$ cannot be submitted to the `Link` oracle. This means that the (in-)equality decisions obtained from the simulated linking proofs are independent of the challenge bit b, while the simulation ensures that the proofs do not contain information about the trapdoor key tk. Taking all together, we obtain an upper bound of the advantage of an adversary in the anonymity game with extensions, i.e., $\Pr[S_0] \leq \mathsf{Adv}_{\mathsf{anon}}(\kappa) + \epsilon_{\mathsf{zk}}(\kappa)$, where $\mathsf{Adv}_{\mathsf{anon}}$ denotes the advantage of an adversary in the plain anonymity game without extensions. □

Linking soundness. It is immediate that the output of an adversary against the linking soundness game contradicts the perfect soundness of the underlying proof system. □

[8] Observe that the zero-knowledge property of Π implies the existence of a simulator.