# Efficient Signatures of Knowledge and DAA in the Standard Model

David Bernhard[1], Georg Fuchsbauer[2,⋆], and Essam Ghadafi[1]

[1] University of Bristol, UK
[2] Institute of Science and Technology, Austria

**Abstract.** Direct Anonymous Attestation (DAA) is one of the most complex cryptographic protocols deployed in practice. It allows an embedded secure processor known as a Trusted Platform Module (TPM) to attest to the configuration of its host computer without violating the owner's privacy. DAA has been standardized by the Trusted Computing Group and ISO/IEC.

The security of the DAA standard and all existing schemes is analyzed in the random-oracle model. We provide the first constructions of DAA in the standard model, that is, without relying on random oracles. Our constructions use new building blocks, including the first efficient signatures of knowledge in the standard model, which have many applications beyond DAA.

**Keywords.** DAA, group signatures, signatures of knowledge, standard model.

## 1 Introduction

Direct Anonymous Attestation (DAA) is a protocol for a secure embedded processor known as a Trusted Platform Module (TPM) to authenticate itself and sign messages attesting to the state of its host while preserving the privacy of its owner. The first DAA protocol by Brickell, Camenisch and Chen [10] was standardized in 2004 by the Trusted Computing Group (TCG) [32] as the TPM 1.2 standard and has since been adopted as an ISO/IEC standard [28]; millions of TPMs have been shipped with personal computers.

In DAA a party owning a TPM can join a group and then sign messages as a member of this group. DAA signatures sign pairs of data, a message and a *basename*, which can be thought of as the identity of the intended verifier. Two signatures on the same basename can be *linked*, that is, they reveal whether they were produced by the same signer. Apart from this, signatures are anonymous; in particular, signatures on different basenames (or empty basenames) hide whether they come from the same user.

Many DAA schemes have been proposed, including [11,16,17,18,19,20], improving both the efficiency of DAA and refining the security model. While the first schemes were analyzed in a simulation-based model, recent papers have switched to game-based models. We prove our results in the most recent model of Bernhard et al. [6] who pointed out shortcomings in the models of some previous papers [10,11,16,17,18,19].

⋆ Work done while at University of Bristol.

M. Jacobson et al. (Eds.): ACNS 2013, LNCS 7954, pp. 518–533, 2013.

The novelty of our schemes is that they are in the *standard model*, where one does not need to rely on the so-called *random-oracle* heuristic [3], which was required in all previous DAA schemes but is problematic in cryptographic theory [13]. It is common practice nowadays to investigate which schemes can be implemented without random oracles. Standard-model schemes are generally less efficient than their random-oracle-based ancestors; we do not intend to improve on the efficiency of earlier DAA schemes but construct efficient schemes in the standard model.

**A Blueprint for DAA.** As argued in [6], all existing DAA schemes follow the same "blueprint" and are constructed from the same building blocks: a Randomizable weakly Blind Signature (RwBS), a Linkable Indistinguishable Tag (LIT) and a Signature of Knowledge (SoK). We discuss these concepts in more detail in Sect. 3 and 4. DAA users hold secret keys, on which they receive a (blind) signature as a certificate from the issuer when joining a group. A DAA signature consists of this certificate, a LIT on the basename under the user's key and a SoK on the message, proving knowledge of a key corresponding to the certificate and the LIT. Our first standard-model DAA scheme largely follows this blueprint; for our second scheme we propose an alternative method of constructing DAA yielding a more efficient scheme.

The security model from [6] operates in two steps: first, the authors discuss *pre-DAA* schemes, which are fully functional DAA schemes but without the option for the TPM to delegate non-security critical operations to its more powerful host computer. Secondly, they give generic methods to perform such delegation securely given a pre-DAA scheme. Since their second step is independent of the random-oracle model (ROM), it is also applicable to our schemes. We therefore restrict ourselves to constructing standard-model pre-DAA schemes in this paper.

**LIT in the Standard Model.** A DAA signature contains a deterministic tag on the basename. This LIT should look random, so tags under different keys are indistinguishable, which is trivially achievable by using a random oracle. Like Verifiable Random Functions (VRF) [30], LITs are much harder to construct in the standard model, in particular, for large input spaces. LITs are somewhat stronger than VRFs, and we do not know of any large-domain VRF which yields a LIT. (See the discussion in Sect. 4.2.)

For DAA, we believe it is reasonable to postulate that the number of possible basenames is polynomial in the security parameter. While the set of *messages* which users can sign must be large, the number of possible verifiers (corresponding to basenames) will be efficiently enumerable.

**Overview of our Paper and Contributions.** In Sect. 2 we introduce some notation as well as the (pre-)DAA definition and security notions from [6].

In Sect. 3 we introduce and construct the first efficient signatures of knowledge [14] without random oracles, which may be of independent interest. SoKs are a generalization of digital signatures and use a witness to an NP statement as the signing key. We build them from Proofs of Knowledge (PoK), of which Groth-Sahai proofs [26] are the only known efficient standard-model instantiation. While the transformation from PoK to SoK is almost trivial in the random-oracle model, Groth-Sahai proofs cannot be used directly since SoKs require strong security properties akin to simulation-sound extractability [25]. Instead, we revert to a known technique, used by Groth [25], to overcome this limitation.

In Sect. 4 we discuss and construct randomizable weakly blind signatures and linkable indistinguishable tags. In order to construct DAA, these building blocks must be compatible with each other and the Groth-Sahai-based SoK. The challenges here are that Groth-Sahai proofs apply only to a limited class of statements and are even more restrictive in security proofs: the language on which we make proofs is that of pairing-product equations [25], in which we can only prove knowledge of elements of a bilinear group. It follows that we have to choose our RwBS and LIT with some care: the RwBS implicitly used in previous DAA schemes, even those that do not require a random oracle directly, are not Groth-Sahai compatible for example. We build on the signature schemes of Abe et al. [1] and Ghadafi [22] to construct different RwBS schemes. As the LIT used in previous schemes is only secure in the ROM, we construct a new LIT based on the VRF by Dodis and Yampolskiy [21].

Using these building blocks, we construct two DAA schemes in Sect. 5 and 6. These are the first DAA schemes in the standard model. Our first construction relies solely on existing, non-interactive assumptions. To improve efficiency, our second construction uses some components from the literature which rely on interactive assumptions.

To evaluate efficiency, we consider the most closely related cryptographic primitive: dynamic group signatures [4], which do not require linkability and handle tracing differently. Our DAA signatures are shorter than Groth's group signatures [24], which is currently the most efficient scheme in the standard model. Moreover, our join protocol involves fewer rounds. See the full version [5] for details.

## 2 Preliminaries

**Notation.** A bilinear group is a tuple $\mathcal{P} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, P_1, P_2)$ where $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ are groups of prime order $p$; $P_1$ and $P_2$ are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively and $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is bilinear (i.e. $e([x]Q_1, [y]Q_2) = e(Q_1, Q_2)^{xy}$ for all $Q_1, Q_2, x$ and $y$) and $e(P_1, P_2)$ generates $\mathbb{G}_T$. All group operations are efficiently computable and $[x]P$ denotes the $x$-fold composition of an element $P$ with itself. We use *asymmetric* bilinear groups (which are more efficient), for which there are no known efficiently computable homomorphisms from $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa. We let $\mathbb{G}^\times := \mathbb{G} \setminus \{0_\mathbb{G}\}$.

**Assumptions.** Our constructions rely on the following assumptions from the literature:

**SXDH.** The DDH assumption holds in both groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

**CDH$^+$ [7].** Given $(P_1, P_2, [a]P_1, [b]P_1, [a]P_2)$, it is hard to compute $[ab]P_1$. This is identical to CDH in symmetric bilinear groups.

**$q$-SDH [8].** Given $(P_1, [x]P_1 \ldots, [x^q]P_1, P_2, [x]P_2)$ for $x \leftarrow \mathbb{Z}_p^\times$, it is hard to output a pair $(c, [\frac{1}{x+c}]P_1) \in \mathbb{Z}_p \times \mathbb{G}_1$ for an arbitrary $c \in \mathbb{Z}_p \backslash \{-x\}$ .

**$q$-DDHI [2].** Given $(P_i, [x]P_i, [x^2]P_i, \ldots, [x^q]P_i)$ where $x \leftarrow \mathbb{Z}_p^\times$ it is hard to distinguish $[\frac{1}{x}]P_i$ from a random element of $\mathbb{G}_i$. Here $i$ can be either 1 or 2.

**$q$-SFP [1].** Given $A, B \in \mathbb{G}_1$, $\tilde{A}, \tilde{B}, G_Z, F_Z, G_R, F_U \in \mathbb{G}_2$, and $q$ random tuples $(Z_i, R_i, S_i, T_i, U_i, V_i, W_i)$ each satisfying $e(A, \tilde{A}) = e(Z_i, G_Z)e(R_i, G_R)e(T_i, S_i)$ and $e(B, \tilde{B}) = e(Z_i, F_Z)e(U_i, F_U)e(W_i, V_i)$, it is hard to output a new such tuple $(Z^*, R^*, S^*, T^*, U^*, V^*, W^*)$ with $Z^* \notin \{Z_i\}_{i=1}^q \cup \{0_{\mathbb{G}_1}\}$.

**DH-LRSW** [22]**.** Given $([x]P_2, [y]P_2)$ for random $(x, y) \leftarrow \mathbb{Z}_p^2$ and an oracle that, on input a *Diffie-Hellman* pair $(M_1, M_2)$ of the form $([m]P_1, [m]P_2)$ for some $m \in \mathbb{Z}_p$, picks a random $a \leftarrow \mathbb{Z}_p$ and outputs a DH-LRSW tuple of the form $([a]P_1, [ay]P_1, [ay]M_1, [ax]P_1 + [axy]M_1)$, it is hard to compute a DH-LRSW tuple for $([m']P_1, [m']P_2)$ that was never queried to the oracle.

**Groth-Sahai Proofs.** Groth-Sahai (GS) proofs [26] are non-interactive proofs in the Common Reference String (CRS) model. We will use GS proofs that are secure under the SXDH assumption (which, as noted by [23], yields their most efficient instantiation) and that prove knowledge of a satisfying assignment for a pairing-product equation

$$\prod e(A_j, \underline{Y_j}) \prod e(\underline{X_i}, B_i) \prod\prod e(\underline{X_i}, \underline{Y_j})^{\gamma_{i,j}} \;=\; \prod e(G_\ell, H_\ell) \qquad (1)$$

(the variables are underlined, all other values are constants). The language for these proofs is of the form $\mathcal{L} := \{\mathsf{statement} \mid \exists\, \mathsf{witness} : E(\mathsf{statement}, \mathsf{witness}) \text{ holds } \}$ where $E(\mathsf{statement}, \cdot)$ is a set of pairing-product equations. The GS proof system is formally defined by a tuple of algorithms

$$(\mathsf{GSSetup}, \mathsf{GSProve}, \mathsf{GSVerify}, \mathsf{GSExtract}, \mathsf{GSSimSetup}, \mathsf{GSSimProve}) \ .$$

$\mathsf{GSSetup}$ takes as input the description of a bilinear group $\mathcal{P}$ and outputs a *binding* reference string $\mathsf{crs}$ and an extraction key $\mathsf{xk}$. $\mathsf{GSProve}$ takes as input $\mathsf{crs}$, a set of equations $\mathsf{statement}$ and a witness, and outputs a proof $\Omega$ for the satisfiability of the equations. We write $\mathsf{GSProve}_{\mathrm{SEC}}(\mathsf{crs}, \{\mathsf{witness}\} : \mathsf{statement} \in \mathcal{L})$, where SEC = ZK means the proofs are zero-knowledge and WI means they are witness-indistinguishable. Given $\mathsf{crs}$, a set of equations and a proof, $\mathsf{GSVerify}$ and outputs 1 if the proof is valid, and else 0.

$\mathsf{GSExtract}$ takes as input a binding $\mathsf{crs}$, the extraction key $\mathsf{xk}$ and a valid proof $\Omega$, and outputs the witness used for the proof. $\mathsf{GSSimSetup}$, on input a bilinear group $\mathcal{P}$, outputs a *hiding* reference string $\mathsf{crs}_{\mathrm{Sim}}$ and a trapdoor key $\mathsf{tr}$ that allows to simulate proofs. $\mathsf{GSSimProve}$ takes $\mathsf{crs}_{\mathrm{Sim}}$, a statement and the trapdoor $\mathsf{tr}$ and produces a simulated proof $\Omega_{\mathrm{Sim}}$ without a witness. The distributions of strings $\mathsf{crs}$ and $\mathsf{crs}_{\mathrm{Sim}}$ are computationally indistinguishable and simulated proofs are indistinguishable from proofs output by $\mathsf{GSProve}$. The proof system has prefect completeness, perfect soundness, composable witness-indistinguishability or composable zero-knowledge. We refer to [26] for the formal definitions.

**Direct Anonymous Attestation: The pre-DAA Model.** The syntax and security model for pre-DAA were defined in [6]. A pre-DAA scheme consists of a tuple of algorithms

$$\big(\mathsf{Setup}, \mathsf{GKg}, \mathsf{UKg}, \langle \mathsf{Join}, \mathsf{Iss} \rangle, \mathsf{GSig}, \mathsf{GVf}, \mathsf{Identify}_{\mathsf{T}}, \mathsf{Identify}_{\mathsf{S}}, \mathsf{Link}\big) \ .$$

$\mathsf{Setup}$, on input the security parameter $1^\lambda$, outputs public parameters $\mathsf{param}$, which is an implicit input to all other algorithms. $\mathsf{GKg}$ outputs $(\mathsf{gmpk}, \mathsf{gmsk})$, a public/secret key pair for the group manager (issuer), and $\mathsf{UKg}$ generates a secret key $\mathsf{sk}$ for a user.

$\langle \mathsf{Join}(\mathsf{sk}_i, \mathsf{gmpk}), \mathsf{Iss}(\mathsf{gmsk}, \mathsf{gmpk}) \rangle$ are the user- and issuer-side procedures for an interactive protocol by means of which a user joins a group. The user takes a secret key $\mathsf{sk}_i$ and the issuer's public key $\mathsf{gmpk}$ as input, whereas the issuer has as input a

key pair $(\mathsf{gmpk}, \mathsf{gmsk})$. If completed successfully, the user obtains a *group signing key* $\mathsf{gsk}$. We assume w.l.o.g. that $\mathsf{gsk}$ contains the issuer's public key $\mathsf{gmpk}$.

$\mathsf{GSig}(\mathsf{gsk}_i, \mathsf{sk}_i, m, \mathsf{bsn})$ is the signing algorithm. It takes as input a group signing key $\mathsf{gsk}_i$, a user secret key $\mathsf{sk}_i$, a message $m$ and a basename $\mathsf{bsn}$ and outputs a DAA signature $\sigma$ on the message $m$ under the basename $\mathsf{bsn}$.

$\mathsf{GVf}(\mathsf{gmpk}, \sigma, m, \mathsf{bsn})$ is the verification algorithm. It returns 1 if the signature $\sigma$ is valid on the message $m$ and the basename $\mathsf{bsn}$ w.r.t. $\mathsf{gmpk}$. Otherwise, it returns 0.

$\mathsf{Identify}_\mathsf{T}(\mathsf{gmpk}, \mathcal{T}, \mathsf{sk}_i)$ is a transcript-tracing algorithm. It is mainly used in the security model although it could be used to trace dishonest users who reveal their secret key. This algorithm takes as input $\mathsf{gmpk}$, a transcript $\mathcal{T}$ of a join/issue protocol execution and a secret key $\mathsf{sk}_i$. It returns 1 if this transcript could have been produced by an honest user with secret key $\mathsf{sk}_i$, and 0 otherwise.

$\mathsf{Identify}_\mathsf{S}(\mathsf{gmpk}, \sigma, m, \mathsf{bsn}, \mathsf{sk}_i)$ is a signature-tracing algorithm. Like $\mathsf{Identify}_\mathsf{T}$, its use is in the security model and possibly to trace dishonest users. On inputs $\mathsf{gmpk}$, a signature $\sigma$, a message $m$, a basename $\mathsf{bsn}$ and a secret key $\mathsf{sk}_i$, it returns 1 iff $\sigma$ could have been produced by an honest user with the secret key $\mathsf{sk}_i$.

$\mathsf{Link}(\mathsf{gmpk}, m_0, \sigma_0, m_1, \sigma_1, \mathsf{bsn})$ is the linking algorithm. Its inputs are $\mathsf{gmpk}$, two messages and signatures $m_0, m_1, \sigma_0, \sigma_1$ and a basename $\mathsf{bsn}$. It returns 1 iff both signatures were produced by the same user on their respective messages and under the non-empty basename $\mathsf{bsn}$.

**Security Definitions of pre-DAA.** Here we provide an informal description of the different security requirements. The formal definitions can be found in [6].

**Correctness:** This demands that signatures produced by honest users are accepted by the verifier, and that the user who produced a valid signature can be traced. Moreover, two signatures by the same user on the same non-empty basename link.

**Anonymity:** An adversary, who may control the group issuer, cannot distinguish which of two users of his choice signed a message as long as he cannot trivially decide this using the linking property.

**Traceability:** No group of users can create an untraceable signature as long as the issuer is honest. (A dishonest issuer could always join untraceable users to his group.) There are two notions of traceability which deal with untraceable signatures and signatures that do not link although they should. Since unlike in group signatures, users do not have public keys corresponding to their secret keys, the group-join transcript is used to identify the user.

**Non-frameability:** No adversary, who may even control the group issuer, can frame an honest user by claiming that this user signed a message he did not sign. There are again two notions: framing a user by creating a signature that traces to his key, or one that links to a previous signature created by that user.

## 3 Efficient Signatures of Knowledge without Random Oracles

Let $\mathcal{L}$ be an *NP language*, defined by a polynomial-time computable relation $R$ as $\mathcal{L} = \{x \mid \exists w : (x, w) \in R\}$. We call $x$ a *statement* in $\mathcal{L}$ and $w$ a *witness* for $x$

if $(x, w) \in R$. A *signature of knowledge* (SoK) for $\mathcal{L}$ consists of the following three algorithms: $\mathsf{SoKSetup}$ takes a security parameter $1^\lambda$ and outputs parameters $\mathsf{par}$. If $(x, w) \in R$ then $\mathsf{SoKSign}(\mathsf{par}, R, x, w, m)$ outputs a signature $\sigma$ on the message $m$ w.r.t. statement $x$. The signature is verified by $\mathsf{SoKVerify}(\mathsf{par}, R, x, m, \sigma)$ outputting 0 or 1. Signatures produced by $\mathsf{SoKSign}$ on inputs $\mathsf{par}$ output by $\mathsf{SoKSetup}$, and any $(R, x, w, m)$ such that $(x, w) \in R$ should be accepted by $\mathsf{SoKVerify}$. The (game-based) security definition for SoK, called *SimExt security* [14] requires the following:

**Simulatability:** There exists a *simulator* which can simulate signatures without having a witness for the statement. It consists of $\mathsf{SoKSimSetup}$ and $\mathsf{SoKSimSign}$: the former outputs parameters together with a *trapdoor* $\mathsf{tr}$ and the latter outputs signatures on input $(\mathsf{par}, \mathsf{tr}, R, x, m)$. It is required that no adversary can distinguish the following two situations: (1) It is given $\mathsf{par}$ output by $\mathsf{SoKSetup}$ and access to a $\mathsf{SoKSign}$ oracle. (2) It is given $\mathsf{par}$ output by $\mathsf{SoKSimSetup}$ and an oracle $\mathsf{SoKSim}$ that on input $(R, x, w, m)$ outputs $\mathsf{SoKSimSign}(\mathsf{par}, \mathsf{tr}, R, x, m)$ if $(x, w) \in R$.

**Extraction:** There exists an algorithm $\mathsf{SoKExtract}$ such that if an adversary, given $\mathsf{par} \leftarrow \mathsf{SoKSimSetup}$ and an oracle $\mathsf{SoKSim}$ as above, outputs a tuple $(R, x, m, \sigma)$, we have: if $\mathsf{SoKVerify}(\mathsf{par}, R, x, m, \sigma) = 1$ and $(R, x, w', m)$, for any $w'$, was never queried to the $\mathsf{SoKSim}$ oracle then $\mathsf{SoKExtract}$ extracts a witness for $x$ from $\sigma$ with overwhelming probability.

Chase and Lysyanskaya [14] offer a generic construction satisfying SimExt security, but it is inefficient due to the use of general Non-Interactive Zero-Knowledge (NIZK) proofs. Our construction is based on Groth-Sahai proofs [26] which are efficient NIZK proofs that do not rely on random oracles but only apply to a restricted language. Our SoKs are thus for the same language, namely *satisfiability of sets of Pairing-Product Equations (PPE)*.

If we generate a binding CRS using $\mathsf{GSSetup}$, we can use $\mathsf{GSExtract}$ to extract a witness from a valid proof. However, in order to simulate GS proofs, we need to set up the CRS via $\mathsf{GSSimSetup}$. In this case proofs become information-theoretically independent of their witnesses, thus we cannot extract anymore. In order to realize simulatability and extractability simultaneously, we revert to a well-known trick that was employed by Groth in the context of PPEs [25]. Our SoK parameters are a binding CRS and a signature-verification key and a SoK is a proof of the following statement: one either knows a witness for the original statement *or* knows a signature on the original statement and the message to be signed, under the key contained in the parameters.

To simulate SoKs, we can now use the corresponding key to sign the statement and the message, and use this signature as a witness for the modified statement. Witness indistinguishability of GS proofs guarantees that simulated SoKs are indistinguishable from SoKs that use the witness of the original statement. Extractability follows since from any SoK we can extract a witness for the *modified* statement. This witness must be for the original statement, as a signature on a statement/message pair which was never signed by the $\mathsf{SoKSim}$ oracle would be a forgery.

**Choosing the Signature Scheme.** As we need to prove knowledge of a signature, we require a scheme whose signatures consist of group elements and whose validity is verified by evaluating PPEs. An ideal candidate would be the signatures by Waters

[33], which are secure under the Computational Diffie-Hellman (CDH) assumption, which is implied by the assumptions required for GS proofs. (Their main drawback is a long public key, which will result in long parameters for our SoK.) Waters signatures are defined over *symmetric* bilinear groups (where $\mathbb{G}_1 = \mathbb{G}_2$). Using the Groth-Sahai instantiation over these groups, our construction yields SoKs for the same statements and under the same assumption as GS proofs. To allow for a more general class of statements, we use the following generalization of Waters signatures to asymmetric groups from [7]:

*Parameter Generation.* Given a bilinear group $\mathcal{P}$, to sign messages of the form $m = (m_1, \ldots, m_N) \in \{0,1\}^N$, choose $(Q, U_0, \ldots, U_N) \leftarrow \mathbb{G}_1^{N+2}$.

*Key Generation.* Choose a secret key $\mathsf{sk} \leftarrow \mathbb{Z}_p$ and set $\mathsf{vk} := [\mathsf{sk}]P_2$.

*Signing.* To sign $(m_1, \ldots, m_N)$ using key $\mathsf{sk}$, choose a random $r \leftarrow \mathbb{Z}_p$ and output

$$\big(W_1 := [\mathsf{sk}]Q + [r](U_0 + \textstyle\sum_{i=1}^{N}[m_i]U_i),\ W_2 := [-r]P_1,\ W_3 := [-r]P_2\big)\ .$$

*Verification.* Check whether $e(W_1, P_2)\, e(U_0 + \sum_{i=1}^{N}[m_i]U_i, W_3) = e(Q, \mathsf{vk})$ and $e(W_2, P_2) = e(P_1, W_3)$.

This scheme is unforgeable under chosen-message attack under the CDH$^+$ assumption. In order to sign arbitrary messages, we assume a collision-resistant hash function $\mathcal{H}\colon \{0,1\}^* \to \{0,1\}^N$ (for a suitable $N$).

**Disjunctions of Pairing-Product Equations.** Groth [25] shows how to express disjunctions of two sets of PPEs as a new set of PPEs. The idea is the following: introduce a "selector equation" of the form $e(P_1, \underline{S} + \underline{T} - P_2) = 1$, which can only be satisfied if either $S$ or $T$ are different from 0. Setting one of them to 0 will enable us to simulate one clause of the disjunction. To do so, for every variable Groth introduces an auxiliary variable and adds an equation.

We choose a more efficient approach inspired by that from [26]. In order to simulate equations of the form (1), it suffices to replace the constants $G_\ell$ by auxiliary variables $G'_\ell$, as then, setting all variables to 0 is a satisfying assignment for (1). Now it only remains to ensure that a signer without the trapdoor is forced to set $G'_\ell$ to $G_\ell$, which is done by adding equations $e(G_\ell - \underline{G'_\ell}, \underline{S}) = 1$, where $S$ can only be set to 0 when the prover knows a signature under the public key from the CRS.

With this intuition in mind we now define our signature of knowledge of a satisfying assignment for a set of pairing-product equations. Regarding the Chase-Lysyanskaya definition, we have fixed the relation $R$ to be the set of all pairs $\big((E_k)_{k=1}^K, ((X_i)_{i=1}^m, (Y_j)_{j=1}^n)\big)$ such that $((X_i), (Y_j)) \in \mathbb{G}_1^m \times \mathbb{G}_2^n$ satisfy $E_k$ for all $1 \le k \le K$.

### 3.1 A Construction of Signatures of Knowledge without Random Oracles

**Setup.** On input $\mathcal{P}$, run $(\mathsf{crs}, \mathsf{xk}) \leftarrow \mathsf{GSSetup}(\mathcal{P})$ and choose parameters $(Q, U_0, \ldots, U_N) \leftarrow \mathbb{G}_1^{N+2}$ and a key pair for Waters signatures: choose $t \leftarrow \mathbb{Z}_p$ and set $T := [t]P_2$. $\mathsf{SoKSetup}$ outputs $\mathsf{par} := (\mathsf{crs}, (Q, U_0, \ldots, U_N, T))$, whereas $\mathsf{SoKSimSetup}$ additionally outputs $(\mathsf{xk}, t)$ as an extraction/simulation trapdoor.

**Signing.** Let $E := (E_k)_{k=1}^K$ be the set of equations representing the statement w.r.t. which we sign, where $E_k$ is

$$\prod_{j=1}^{n} e(A_{k,j}, \underline{Y_j}) \prod_{i=1}^{m} e(\underline{X_i}, B_{k,i}) \prod_{i=1}^{m}\prod_{j=1}^{n} e(\underline{X_i}, \underline{Y_j})^{\gamma_{k,i,j}} = \prod_{\ell=1}^{M_k} e(G_{k,\ell}, H_{k,\ell}) \ , \quad (E_k)$$

and let $((X_i)_{i=1}^m, (Y_j)_{j=1}^n)$ be a witness for $E$. We define a new set of equations $E'$:

*(i) Modified equations.* For all $1 \le k \le K$:

$$\prod e(A_{k,j}, \underline{Y_j}) \prod e(\underline{X_i}, B_{k,i}) \prod e(\underline{G'_{k,\ell}}, -H_{k,\ell}) \prod\prod e(\underline{X_i}, \underline{Y_j})^{\gamma_{k,i,j}} = 1 \ .$$

*(ii) Selector equations.* For all $1 \le k \le K, 1 \le \ell \le M_k$: $e(G_{k,\ell} - \underline{G'_{k,\ell}}, T - \underline{T'}) = 1$.
*(iii) Signature-verification equations.*

$$e(\underline{W_1}, P_2)\, e(U_0 + \textstyle\sum_{i=1}^{N} [h_i] U_i, \underline{W_3}) = e(Q, \underline{T'}) \qquad e(\underline{W_2}, P_2) = e(P_1, \underline{W_3})$$

To sign a message $m \in \{0,1\}^*$ under $\mathsf{par} := (\mathsf{crs}, (Q, U_0, \ldots, U_N, T))$ for the statement $E$ using witness $((X_i), (Y_j))$ proceed as follows:

- Set $T' = W_1 = W_2 = W_3 := 0$ and $G'_{k,\ell} := G_{k,\ell}$, for all $k$ and $\ell$.
- Compute $h = (h_1, \ldots, h_N) := \mathcal{H}(E \| m) \in \{0,1\}^N$, where $E$ is an encoding of the original equations.
- The SoK is a GS proof $\Sigma$ of satisfiability of the set of equations $E'$, using as witness

$$(T', W_1, W_2, W_3, X_1, \ldots, X_m, Y_1, \ldots, Y_n, G'_{1,1}, \ldots, G'_{K,M_K}) \ . \qquad (2)$$

**Verification.** Under $\mathsf{par} := (\mathsf{crs}, (Q, U_0, \ldots, U_N, T))$, to verify a SoK $\Sigma$ on $m$ for the statement $E$, verify that under $\mathsf{crs}$ the GS proof $\Sigma$ is valid on the statement $E'$ for the values $A_{k,j}, B_{k,i}, G_{k,\ell}, H_{k,\ell}$ and $\gamma_{k,i,j}$ from the description of $E$, values $T$ and $(Q, U_0, \ldots, U_N)$ from $\mathsf{par}$ and $h$ defined as $\mathcal{H}(E \| m)$.

**Theorem 1.** *The above is a signature-of-knowledge scheme satisfying SimExt security for the language of sets of pairing-product equations.*

*Proof sketch.* To simulate a signature without knowing a witness, one uses the trapdoor $t$ to make a signature $(W_1, W_2, W_3)$ on $(h_1, \ldots, h_N) := \mathcal{H}(E \| m)$, and sets $T' := T$ and all remaining witnesses components $X_i = Y_j = G'_{k,\ell} := 0$, which satisfies $E'$. Simulatability then follows from witness indistinguishability of GS proofs.

For "Extraction", consider an adversary that has never queried a signature for a pair $(E, m)$, but outputs a SoK $\Sigma$ for it. By soundness of GS proofs, we can extract from $\Sigma$ a witness for $E'$ of the form (2). We must have $T' \neq T$, as otherwise $(W_1, W_2, W_3)$ would be a forgery on $(E \| m)$ (which was never queried to the simulator) by equations (iii) of $E'$. Together with equations (ii) of $E'$, $T' \neq T$ implies that $G'_{k,\ell} = G_{k,\ell}$ for all $k, \ell$, and therefore, by (i), $((X_i), (Y_j))$ is a witness for the original equation $E$. We have thus extracted a witness for $E$. □

To reduce the parameter length, but relying on stronger ("$q$-type") assumptions, we could replace Waters signatures with any of the structure-preserving schemes from [1].

# 4 New Building Blocks

## 4.1 Randomizable Weakly Blind Signatures

Bernhard et al. [6] introduce Randomizable Weakly Blind Signatures (RwBS) as one of the building blocks for DAA. These are similar to blind signatures [15,31] except that blindness must hold only against adversaries that never get to see the message they signed, that is, a signature should not be linkable to its issuing session.

Randomizability means that given a signature, anyone can produce a fresh signature on the same message. We construct two RwBS that do not rely on random oracles. The syntax and security definitions for RwBS can be found in [6,5].

**Partially Randomizable Weakly Blind Signatures.** To work with our SoKs, we require our scheme to be *structure-preserving*: the signatures and the messages it signs must be group elements and the verification equations must be pairing-product equations. For our first construction we use a standard-model signature scheme based on non-interactive assumptions by Abe et al. [1], which we call AHO after its authors. Its security relies on the $q$-SFP assumption (see Sect. 2). Abe et al. show that six of the seven group elements which constitute an AHO signature can be randomized. (We are not aware of a fully randomizable structure-preserving scheme based on non-interactive assumptions.)

This randomizability is useful, since we show that if the signer is given parts of a (partial) randomization of a signature he issued earlier, they are independent of the original signature. When used as a certificate for DAA, we thus only need to hide part of the certificate in a DAA signature to guarantee anonymity. We therefore further relax the notion of weak blindness from [6] to *partial weak blindness* defined w.r.t. a projection function $\pi$. In the security game a signer blindly signs a message chosen by the game. He is then either given the projection of a (partial) randomization of his signature or of a signature on another message and should not be able to distinguish the two cases. The details of this notion and our construction can be found in the full version [5].

**Fully Randomizable Weakly Blind Signatures.** In order to provide a more efficient DAA scheme, we construct a RwBS satisfying the original definition of [6]. Our construction uses a fully randomizable signature scheme by Ghadafi [22] called NCL, which is a structure-preserving variant of CL-signatures [12] based on a variant of the LRSW assumption [29] (see Sect. 2).

Messages of NCL are of the form $([m]P_1, [m]P_2) \in \mathbb{G}_1 \times \mathbb{G}_2$ for some $m \in \mathbb{Z}_p$. The secret and verification keys are of the form $(x, y) \in \mathbb{Z}_p^2$ and $([x]P_2, [y]P_2)$, respectively. A message $(M_1, M_2)$ is signed by choosing a random $a \leftarrow \mathbb{Z}_p^\times$ and outputting

$$\big(A := [a]P_1,\ B := [y]A, C := [ay]M_1,\ D := [x](A + C)\big)\ .$$

The verification equations are $A \neq 0$, $e(B, P_2) = e(A, Y)$, $e(C, P_2) = e(B, M_2)$, $e(D, P_2) = e(A, X)\, e(C, X)$ and $e(M_1, P_2) = e(P_1, M_2)$. A signature is randomized by choosing $a' \leftarrow \mathbb{Z}_p^\times$ and setting $A' := [a']A$, $B' := [a']B$, $C' := [a']C$, $D' := [a']D$.

We observe that to compute a signature on a message $(M_1, M_2)$, only $M_1$ is required, whereas verification of the signature could be done using $M_2$ only. A first idea to construct a weakly blind signature from NCL is to define BSRequest as only sending

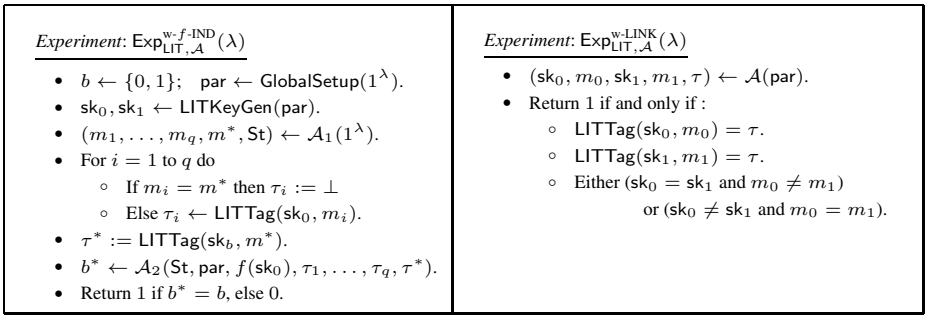


**Fig. 1.** Security games for indistinguishability (left) and linkability (right) of LIT.

$M_1$. However, in the security proof of weak blindness, the simulator (playing the user) will not have $M_2$ (otherwise it could break the notion itself) and can therefore not verify the correctness of the adversary's signature. We therefore require the signer to provide a NIZK proof of correctness of the signature.

Moreover, in the reduction of blind-signature unforgeability to unforgeability of NCL, the simulator (playing the signer) needs the full message $(M_1, M_2)$ to query its signing oracle. Therefore, when requesting a signature, the user must provide a NIZK proof of knowledge of $M_2$. These NIZK proofs use different CRSs (as the reductions exploit different properties) and are efficiently implemented using Groth-Sahai proofs.

We refer to the full version [5] for the details of our scheme and a security proof.

### 4.2 Linkable Indistinguishable Tags

The second building block introduced to construct DAA schemes generically in [6] is a *Linkable Indistinguishable Tag* (LIT). These tags are similar to MACs, but have stronger security requirements. LIT schemes are defined w.r.t. a one-way function $\mathsf{PK}()$ such that a tag created with a secret key $\mathsf{sk}$ can be verified given $\mathsf{PK}(\mathsf{sk})$ rather than $\mathsf{sk}$. Thus, $\mathsf{PK}(\mathsf{sk})$ can be viewed as a public key for the tag. A LIT scheme is defined by the following algorithms. W.l.o.g. we assume that there is an algorithm $\mathsf{GlobalSetup}$ which generates global parameters $\mathsf{par}$ (such as a bilinear group), which all algorithms take as an (implicit) input.

$\mathsf{LITKeyGen}(\mathsf{par})$ takes global parameters $\mathsf{par}$ and outputs a secret key $\mathsf{sk}$.

$\mathsf{LITTag}(\mathsf{sk}, m)$ is deterministic, takes as input a secret key $\mathsf{sk}$ and a message $m$, and outputs a tag $\tau$.

$\mathsf{LITVerify}(\mathsf{PK}(\mathsf{sk}), m, \tau)$ is given the image of $\mathsf{sk}$ under $\mathsf{PK}$, a message $m$ and a tag $\tau$ and checks whether $\tau$ is a valid tag on the message $m$ w.r.t. $\mathsf{sk}$, outputting 1 or 0.

**Security.** Besides correctness, [6] defines the notions linkability and indistinguishability, of which we only require relaxations. A LIT is *weakly linkable* if the following holds: if two tags are identical then they are either w.r.t. the same key and the same message, or both keys *and* both messages are different. In particular, two tags

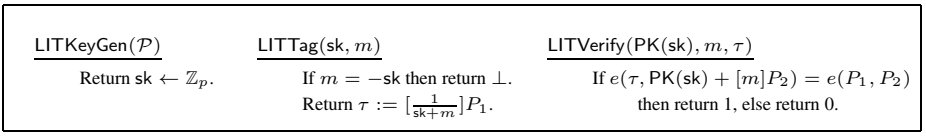


**Fig. 2.** The WBB-based Linkable Indistinguishable Tag (WBB-LIT).

under different keys on the same message (or under the same key on different messages) must be different. Weak linkability was used implicitly in [6]. We formalize it by experiment $\mathsf{Exp}^{\text{w-LINK}}_{\mathsf{LIT},\mathcal{A}}(\lambda)$ in Fig. 1 and say a LIT scheme is *weakly linkable* if $\mathsf{Adv}^{\text{w-LINK}}_{\mathsf{LIT},\mathcal{A}}(\lambda) := \Pr[\mathsf{Exp}^{\text{w-LINK}}_{\mathsf{LIT},\mathcal{A}}(\lambda) = 1]$ is negligible in $\lambda$ for any PPT adversary $\mathcal{A}$.

The LIT *$f$-indistinguishability* is defined w.r.t. a one-way function $f$ and states that no adversary, having access to a $\mathsf{LITTag}(\mathsf{sk}, \cdot)$ oracle, can distinguish a tag on a message of his choice (for which he did not query the oracle) from a tag produced under a different random key. This should hold even if the adversary is given the image $f(\mathsf{sk})$ of the secret key in question. We weaken this property by requiring that the adversary submit all the oracle queries and announce the message to be challenged on *before* seeing the parameters and the image of the secret key. This is formalized by $\mathsf{Exp}^{\text{w-}f\text{-IND}}_{\mathsf{LIT},\mathcal{A}}(\lambda)$ in Fig. 1 and we say a LIT scheme is *weakly $f$-indistinguishable* if $\mathsf{Adv}^{\text{w-}f\text{-IND}}_{\mathsf{LIT},\mathcal{A}}(\lambda) := \left|2 \cdot \Pr[\mathsf{Exp}^{\text{w-}f\text{-IND}}_{\mathsf{LIT},\mathcal{A}}(\lambda) = 1] - 1\right|$ is negligible in $\lambda$ for any PPT adversary $\mathcal{A}$.

**Small Message Spaces.** If the size of the message space is polynomial in the security parameter then $f$-indistinguishability from [6] is implied by its weak version: assuming an adversary $\mathcal{A}$ breaking the standard notion, we can construct an adversary $\mathcal{B}$ breaking the weak notion as follows: Let $\{m_1, \ldots, m_\ell\}$ be the message space, with $\ell = poly(\lambda)$. Then $\mathcal{B}$ randomly picks $i \leftarrow \{1, \ldots, \ell\}$ and outputs its queries and the challenge as $(m_1, \ldots, m_{i-1}, m_{i+1}, \ldots, m_\ell, m^* := m_i)$. With non-negligible probability $\mathcal{A}$ will ask to be challenged on $m_i$, in which case $\mathcal{B}$ can simulate all Tag queries and use $\mathcal{A}$ to break weak $f$-indistinguishability.

**A Weak LIT in the Standard Model.** The weak Boneh-Boyen signature scheme [8] was used in [21,2] to construct verifiable random functions [30] for small message spaces under two variants of the DDHI assumption. The proof of pseudorandomness uses a technique similar to that for the unforgeability of weak Boneh-Boyen signatures in [8]: if the queried messages $m_1, \ldots, m_n$ and the challenge $m^*$ (whose VRF value is to be distinguished from random) are known in advance then given a DDHI instance, we can set up the VRF parameters and the public key so that we can (1) construct the VRF values on $m_1, \ldots, m_n$ and (2) use the DDHI challenge to construct a challenge for $m^*$. Using the proof strategy for small message spaces discussed above, this suffices to prove pseudorandomness. We define our LIT as the VRF from [2] and use the first part of their proof of pseudorandomness of VRFs to prove weak $f$-indistinguishability w.r.t. $f(\mathsf{sk}) := [\mathsf{sk}]P_1$.

**Theorem 2.** *The* WBB-LIT *in Fig. 2 is a LIT for* $\mathsf{PK}(\mathsf{sk}) := [\mathsf{sk}]P_2$. *It is weakly linkable and satisfies weak $f$-indistinguishability for $f(\mathsf{sk}) := [\mathsf{sk}]P_1$ if the DDHI assumption holds in group $\mathbb{G}_1$.*

Since it is impossible to have $\tau = \mathsf{LITTag}(\mathsf{sk}_0, m_0) = \mathsf{LITTag}(\mathsf{sk}_1, m_1)$ with ($\mathsf{sk}_0 = \mathsf{sk}_1$ and $m_0 \neq m_1$) or ($\mathsf{sk}_0 \neq \mathsf{sk}_1$ and $m_0 = m_1$), weak linkability holds unconditionally. Weak $f$-indistinguishability is proved analogously to the pseudorandomness property of the VRF under the $q$-DDHI assumption (see [21, Theorem 2]).

**LIT vs. VRF.** To construct a LIT which is fully indistinguishable *and* supports large domains, a natural approach would be to consider large-domain VRFs. There have been several such schemes in the recent literature, e.g. [9,2,27]. Unfortunately, all of the aforementioned schemes violate the weak linkability requirement for LITs, as it is easy to find $\mathsf{sk}_0 \neq \mathsf{sk}_1$ and $m$ such that $\mathsf{LITTag}(\mathsf{sk}_0, m) = \mathsf{LITTag}(\mathsf{sk}_1, m)$. While in the random-oracle model LITs are easy to construct, it is not clear how to construct fully indistinguishable LITs for large basename spaces without resorting to interactive assumptions. VRFs are already hard to construct, but due to the subtle linkability requirement, LITs seem even harder.

## 5 A Generic Construction of pre-DAA in the Standard Model

Our first construction of pre-DAA uses AHO signatures as partially weakly blind signatures, the VRF from [2] given in Fig. 2 as a LIT and Waters signatures [33] implicitly for the signatures of knowledge, which are themselves Groth-Sahai proofs [26].

The setup algorithm outputs a bilinear group and parameters for the SoK. The issuer generates an AHO signature key pair as $(\mathsf{gmsk}, \mathsf{gmpk})$. To join a group, a user creates a LIT key $\mathsf{sk}$ and sends $F_1 := [\mathsf{sk}]P_1$ to the issuer, who replies with an AHO signature $\mathsf{cred}$ on $F_1$. To make a DAA signature on a message $m$ under a basename $\mathsf{bsn}$, a user first (partially) randomizes his AHO signature $\mathsf{cred}$ and then splits it into a public part $\mathsf{cred}_P$ and a part $\mathsf{cred}_H$ which he will include in the witness for the SoK. Next, he creates a LIT tag $\tau := \mathsf{LITTag}(\mathsf{sk}, \mathsf{bsn})$ on the basename using his key. He then computes a signature of knowledge $\Sigma$ on the message $\mathsf{bsn}\|m$ proving knowledge of a LIT key $\mathsf{sk}$ and the hidden part of an AHO signature $\mathsf{cred}_H$ such that the tag and the AHO signature both verify w.r.t. this key. The DAA signature is $\sigma := (\mathsf{cred}_P, \tau, \Sigma)$.

We formalize the above. The language of the SoK needs to be a pairing-product equation as in (1) with witnesses in $\mathbb{G}_1$ and $\mathbb{G}_2$. Rather than proving knowledge of $\mathsf{sk}$, the witness will be $F_1 := [\mathsf{sk}]P_1$ and $F_2 := [\mathsf{sk}]P_2$. The AHO signature is on $F_1$ rather than $\mathsf{sk}$ so $F_1$ is also sufficient to verify it. The signature is split into a public part $\mathsf{cred}_P := (S, T, V, W)$ and a hidden part $\mathsf{cred}_H = (Z, R, U)$ (see the full version [5] for the details). We let $\mathsf{BSVerify}'(\mathsf{gmpk}, F_1, (\mathsf{cred}_H, \mathsf{cred}_P))$ denote the AHO verification algorithm of a split signature on $F_1$.

The value $F_2$ is the public key for the LIT from Sect. 4.2, so $\tau$ can be verified using $\mathsf{LITVerify}(F_2, \mathsf{bsn}, \tau)$. It remains to prove that $(F_1, F_2)$ is a Diffie-Hellman pair, that is, of the form $([\mathsf{sk}]P_1, [\mathsf{sk}]P_2)$. The language of the SoK is thus

$$\mathcal{L}: \Big\{\big((\mathsf{gmpk}, \mathsf{cred}_P, \mathsf{bsn}, \tau), (F_1, F_2, \mathsf{cred}_H)\big) \;:\; e(-P_1, \underline{F_2})\, e(\underline{F_1}, P_2) = 1 \\ \wedge\; \mathsf{BSVerify}'(\mathsf{gmpk}, \underline{F_1}, (\underline{\mathsf{cred}_H}, \mathsf{cred}_P)) = 1 \wedge\; \mathsf{LITVerify}(\underline{F_2}, \mathsf{bsn}, \tau) = 1\Big\}\ .$$

If $\mathsf{bsn} = \perp$ then the DAA signature is $(\mathsf{cred}_P, \Sigma)$, where $\Sigma$ is a SoK for the language $\mathcal{L}' : \big\{((\mathsf{gmpk}, \mathsf{cred}_P), (F_1, \mathsf{cred}_H)) \;:\; \mathsf{BSVerify}'(\mathsf{gmpk}, \underline{F_1}, (\underline{\mathsf{cred}_H}, \mathsf{cred}_P)) = 1\big\}$.

To verify a DAA signature, one verifies the SoK. To link two signatures under the same basename, one compares the contained tags $\tau$ and returns 1 if they are equal; to identify a transcript given $\mathsf{sk}$, one checks if the first message by the user is the value $[\mathsf{sk}]P_1$; and to identify a signature, one checks the LIT tag $\tau$ using $\mathsf{sk}$ and $\mathsf{bsn}$. Our construction follows closely the blueprint from [6] (except for proving knowledge of a function of $\mathsf{sk}$ and hiding parts of the certificate) and is proven analogously.

## 6 A More Efficient pre-DAA Scheme in the Standard Model

To construct a truly efficient pre-DAA scheme, we replace the *partially* randomizable AHO-based *partially* weak blind signatures by the fully randomizable NCL-based RwBS; this avoids having to include parts of the certificate in the SoK. In addition, we replace the SoKs by more efficient Proofs of Knowledge (PoK).

The main obstacle in doing so is that the user secret key $\mathsf{sk}$ is used both for the tag on $\mathsf{bsn}$ and (implicitly in the SoK) for the signature on the message $m$. Suppose we replaced the SoK of $\mathsf{sk}$ by a regular Groth-Sahai proof of knowledge of $\mathsf{sk}$ and of a signature on $m$. Non-frameability corresponds to a forgery of a signature on $m$, to which the notion must be reduced. In the reduction we thus have to extract a signature from the PoK, and therefore cannot simulate proofs, as GS proofs only allow extraction *or* simulation (while SimExt security of SoKs allows both at the same time.) However, if we do not simulate the PoK then when answering DAA-signing queries, we need to provide actual tags—for which we do not have the user's secret key.

We overcome this by using a novel approach: we use a signature scheme which in the reduction allows us to simulate tags under the same secret key and a tag scheme which allows us to simulate signatures. We do so by choosing the schemes in a way that tags of one scheme and signatures of the other scheme have the same form—although the security requirements are different, and they are based on different assumptions. In particular, note that the values of the VRF from [2] are essentially "weak" Boneh-Boyen signatures [8]. (These signatures are only secure against adversaries which make all signing queries before seeing the public key.) Weak signatures can easily be turned into standard signatures using a hybrid construction, where one signs a verification key of a one-time signature and uses the corresponding secret key to sign the actual message. Unlike for the message space of the LIT (i.e. the basename space), there is no restriction on the message spaces of the signature schemes (and thus the message space of our DAA is big enough to sign messages of arbitrary length by hashing them first).

We separate the domains for the messages of the weak signatures and the messages of the tags by prepending a bit to the messages. In the reduction of non-frameability to weak signature unforgeability we can then use our (weak) signing oracle to obtain signatures *and* simulate the tags: The basename space is polynomial in size and the verification keys of the one-time signatures can be produced beforehand; we can therefore make our signature queries on all basenames and on the one-time keys beforehand.

Then, in the proof of anonymity we use the trick the other way round and simulate signatures using the tag oracle. In the reduction to weak $f$-indistinguishability of the tags, we can again make all tag queries (on basenames and one-time verification keys) upfront. Another advantage of this approach is that, since weak signatures have the form of LITs, they are unlinkable to the key that produced them, which means that we can

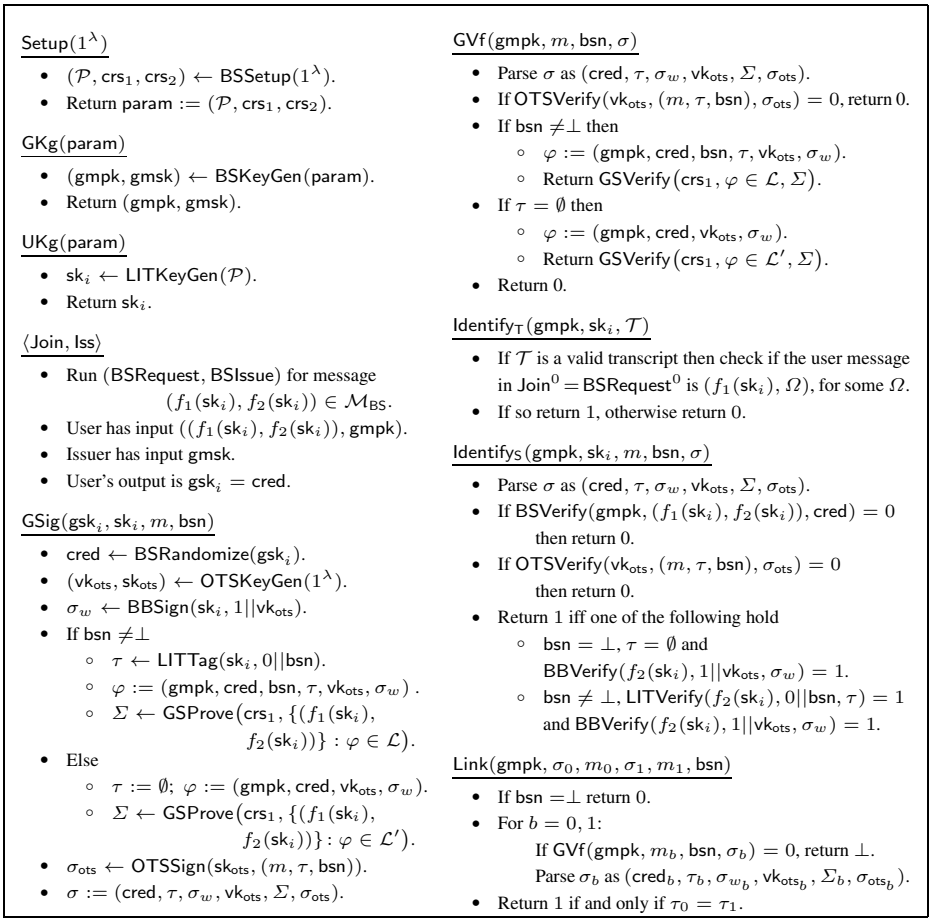


**Fig. 3.** An efficient pre-DAA scheme construction in the standard model

even include the weak signatures in the clear in the DAA; we thus only need to prove knowledge of the secret key.

Our construction is shown in Fig. 3 and uses the LIT scheme from Fig. 2 and the NCL-based RwBS described in Sect. 4.1. As in the generic scheme, the user group signing key gsk is a credential (i.e. a blind signature) obtained from the issuer when joining the group. To make a DAA signature, the user randomizes gsk to cred, chooses a one-time signature key pair $(\mathsf{sk}_{\mathsf{ots}}, \mathsf{vk}_{\mathsf{ots}})$ and uses his secret key sk to generate a LIT tag $\tau$ on $0\|\mathsf{bsn}$ (if $\mathsf{bsn} = \perp$ then $\tau := \emptyset$), and a weak signature $\sigma_w$ on $1\|\mathsf{vk}_{\mathsf{ots}}$. The user then produces a GS PoK $\Sigma$ of $(f_1(\mathsf{sk}) := [\mathsf{sk}]P_1, f_2(\mathsf{sk}) := [\mathsf{sk}]P_2)$ showing well-formedness, that cred is valid on it and that $\tau$ and $\sigma_w$ both verify under $f_2(\mathsf{sk})$.

The DAA signature is defined as $\sigma := (\mathsf{cred}, \tau, \sigma_w, \mathsf{vk}_{\mathsf{ots}}, \Sigma, \sigma_{\mathsf{ots}})$, where $\sigma_{\mathsf{ots}}$ is a one-time signature produced with $\mathsf{sk}_{\mathsf{ots}}$ on the tuple $(m, \tau, \mathsf{bsn})$. Note that the one-time signature also only needs to be weakly unforgeable, as the message $(m, \tau, \mathsf{bsn})$ is known before $\mathsf{vk}_{\mathsf{ots}}$ is chosen. The languages for the GS proofs are defined as follows, where $\mathcal{L}'$ is used when $\mathsf{bsn} = \perp$ and $\mathcal{L}$ otherwise.

$$\mathcal{L}: \big\{\big((\mathsf{gmpk}, \mathsf{cred}, \mathsf{bsn}, \tau, \mathsf{vk}_{\mathsf{ots}}, \sigma_w), (F_1, F_2)\big) : e(-P_1, \underline{F_2})e(\underline{F_1}, P_2) = 1 \\ \wedge\ \mathsf{BSVerify}(\mathsf{gmpk}, (\underline{F_1}, \underline{F_2}), \mathsf{cred}) = 1 \wedge\ \mathsf{LITVerify}(\underline{F_2}, 0||\mathsf{bsn}, \tau) = 1 \\ \wedge\ \mathsf{BBVerify}(\underline{F_2}, 1||\mathsf{vk}_{\mathsf{ots}}, \sigma_w) = 1\big\}$$

$$\mathcal{L}': \big\{\big((\mathsf{gmpk}, \mathsf{cred}, \mathsf{vk}_{\mathsf{ots}}, \sigma_w), (F_1, F_2)\big) : e(-P_1, \underline{F_2})e(\underline{F_1}, P_2) = 1 \\ \wedge\ \mathsf{BSVerify}(\mathsf{gmpk}, (\underline{F_1}, \underline{F_2}), \mathsf{cred}) = 1 \wedge\ \mathsf{BBVerify}(\underline{F_2}, 1||\mathsf{vk}_{\mathsf{ots}}, \sigma_w) = 1\big\}$$

A detailed analysis of the efficiency of the construction can be found in the full version [5], where we also give a proof of the following.

**Theorem 3.** *If the NCL-based RwBS scheme is unforgeable and weakly blind, the LIT scheme is weakly linkable and weakly $f$-indistinguishable, the Groth-Sahai proof system is sound and zero-knowledge, and the one-time signature scheme is weakly unforgeable then the construction in Fig. 3 is a secure pre-DAA scheme.*

**Acknowledgements.** This work was supported by ERC Advanced Grant ERC-2010-AdG-267188-CRIPTO, EPSRC via grant EP/H043454/1 and the European Commission through the ICT Programme under Contract ICT2007216676 ECRYPT II.

## References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (2010)
2. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Compact e-cash and simulatable VRFs revisited. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 114–131. Springer, Heidelberg (2009)
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS 1993, pp. 62–73. ACM (1993)
4. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005)
5. Bernhard, D., Fuchsbauer, G., Ghadafi, E.: Efficient signatures of knowledge and DAA in the standard model. Cryptology ePrint Archive. Report 2012/475, `http://eprint.iacr.org/2012/475`
6. Bernhard, D., Fuchsbauer, G., Ghadafi, E., Smart, N.P., Warinschi, B.: Anonymous attestation with user-controlled linkability. Cryptology ePrint Archive. Report 2011/658, `http://eprint.iacr.org/2011/658`
7. Blazy, O., Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Signatures on randomizable ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 403–422. Springer, Heidelberg (2011)
8. Boneh, D., Boyen, X.: Short Signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology 21(2), 149–177 (2008)
9. Boneh, D., Montgomery, H.W., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: CCS 2010, pp. 131–140. ACM (2010)
10. Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: CCS 2004, pp. 132–145. ACM (2004)
11. Brickell, E., Chen, L., Li, J.: Simplified security notions for direct anonymous attestation and a concrete scheme from pairings. Int. Journal of Information Security 8, 315–330 (2009)

12. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
13. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: STOC 1998, pp. 209–218. ACM (1998)
14. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 78–96. Springer, Heidelberg (2006)
15. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO 1982, pp. 199–203. Plenum Press (1983)
16. Chen, L.: A DAA scheme requiring less TPM resources. In: Bao, F., Yung, M., Lin, D., Jing, J. (eds.) Inscrypt 2009. LNCS, vol. 6151, pp. 350–365. Springer, Heidelberg (2010)
17. Chen, L., Morrissey, P., Smart, N.P.: Pairings in trusted computing. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 1–17. Springer, Heidelberg (2008)
18. Chen, L., Morrissey, P., Smart, N.P.: On proofs of security for DAA schemes. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 156–175. Springer, Heidelberg (2008)
19. Chen, L., Morrissey, P., Smart, N.P.: DAA: Fixing the pairing based protocols. Cryptology ePrint Archive. Report 2009/198, http://eprint.iacr.org/2009/198
20. Chen, X., Feng, D.: Direct anonymous attestation for next generation TPM. Journal of Computers 3, 43–50 (2008)
21. Dodis, Y., Yampolskiy, A.: A verifiable random function with short proofs and keys. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 416–431. Springer, Heidelberg (2005)
22. Ghadafi, E.: Formalizing group blind signatures and practical constructions without random oracles. In: Cryptology ePrint Archive, Report 2011/402, http://eprint.iacr.org/2011/402.pdf
23. Ghadafi, E., Smart, N.P., Warinschi, B.: Groth–Sahai proofs revisited. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 177–192. Springer, Heidelberg (2010)
24. Groth, J.: Fully anonymous group signatures without random oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007)
25. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006)
26. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
27. Hohenberger, S., Waters, B.: Constructing verifiable random functions with large input spaces. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 656–672. Springer, Heidelberg (2010)
28. International Organisation for Standardisation (ISO). ISO/IEC 11889: Information technology – Trusted Platform Module (2009)
29. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems (Extended abstract). In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000)
30. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: FOCS 1999, pp. 120–130. IEEE Computer Society (1999)
31. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology 13(3), 361–396 (2000)
32. Trusted Computing Group. TCG TPM specification 1.2 (2003), http://www.trustedcomputinggroup.org
33. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)