Hierarchical Attribute-Based Signatures: Short Keys and Optimal Signature Length

Daniel Gardham(✉) and Mark Manulis

Surrey Centre for Cyber Security, University of Surrey, Guildford, UK
d.gardham@surrey.ac.uk, mark@manulis.eu

Abstract. With Attribute-based Signatures (ABS) users can simultaneously sign messages and prove compliance of their attributes, issued by designated attribute authorities, with some verification policy. Neither signer's identity nor possessed attributes are leaked during the verification process, making ABS schemes a handy tool for applications requiring privacy-preserving authentication. Earlier ABS schemes lacked support for hierarchical delegation of attributes (across tiers of attribute authorities down to the signers), a distinct property that has made traditional PKIs more scalable and widely adoptable.

This changed recently with the introduction of Hierarchical ABS (HABS) schemes, where support for attribute delegation was proposed in combination with stronger privacy guarantees for the delegation paths (path anonymity) and new accountability mechanisms allowing a dedicated tracing authority to identify these paths (path traceability) and the signer, along with delegated attributes, if needed. Yet, current HABS construction is generic with inefficient delegation process resulting in suboptimal signature lengths of order $O(k^2|\Psi|)$ where Ψ is the policy size and k the height of the hierarchy.

This paper proposes a direct HABS construction in bilinear groups that significantly improves on these bounds and satisfies the original security and privacy requirements. At the core of our HABS scheme is a new delegation process based on the length-reducing homomorphic trapdoor commitments to group elements for which we introduce a new delegation technique allowing step-wise commitments to additional elements without changing the length of the original commitment and its opening. While also being of independent interest, this technique results in shorter HABS keys and achieves the signature-length growth of $O(k|\Psi|)$ which is optimal due to the path-traceability requirement.

1 Introduction

Attribute-based Signatures, first introduced in [30] and [31], provide privacy-preserving mechanisms for authenticating messages. An ABS signature assures the verifier that the signer owns a set of attributes that satisfy the signing policy without leaking their identity, nor the set of attributes used. Traditional

R. H. Deng et al. (Eds.): ACNS 2019, LNCS 11464, pp. 89–109, 2019.
https://doi.org/10.1007/978-3-030-21568-2_5

ABS schemes considered two security properties, user privacy and unforgeability. Informally, a user is anonymous if an ABS signature does not leak their identity, nor the set of attributes used to satisfy the signing policy, while unforgeability requires that a signer cannot produce a signature conforming to a policy for which he does not own a set of suitable attributes. Later constructions [14,16,20] offered more advanced functionality with an additional property of traceability which holds signers accountable by allowing a dedicated tracing authority to identity them if required.

The vast majority of existing ABS schemes [5,13,14,16,20,32,33,35] are non-interactive, in the standard model and is based on bilinear maps and Groth-Sahai proofs [21], with the exception of [23], which uses RSA setting and the random oracle model, and the recent schemes in [15,38] which rely on lattices. Interactive ABS schemes, e.g. [27], where policies must be chosen by verifiers ahead of the signing phase have also been proposed. In general, signing polices can have varying levels of flexibility and range from threshold policies [30], to monotone boolean predicates [14,20], and generalised circuits [35]. Typically, more restrictive policies allow for more efficient constructions. Policy-based Signatures (PBS) [5] can be viewed as a generalisation of ABS schemes, albeit their security is currently proven in a single-user setting without addressing stronger non-frameability requirement of more recent ABS schemes [12,14,20].

Hierarchical Attribute-Based Signature and Their Limitations. Hierarchical Attribute-based Signatures (HABS), recently introduced in [12], extend traditional ABS schemes by permitting controlled delegation of attributes from a root authority (RA) over possibly multiple intermediate authorities (IAs) down to the users. In this way HABS aims to close the gap between ABS and traditional PKIs where hierarchical delegation can be achieved at low cost. In HABS, IAs can delegate attributes to any authority in the scheme and users can acquire attributes from any authority in the hierarchy that is authorised to issue them. In addition to strong non-frameability property in a multi-user setting, the authors extend traditional ABS privacy guarantees to protect not only the identity of the signer but also the identities of all intermediate authorities in the delegation path, as part of their new path-anonymity property. Traditional traceability property of ABS schemes has also been extended to hold not only signers but also intermediate authorities accountable for their actions, through the new notion of path-traceability where a dedicated tracing authority can reveal the entire delegation path, along with delegated attributes.

We observe that the HABS scheme in [12] is generic, based on standard cryptographic primitives, i.e., public key encryption, one-time signature, tag-based signature, and non-interactive zero-knowledge proofs. Its delegation process is handled using a tag-based signature (TBS) where an authority at level i produces a TBS signature, using attribute as a tag, on the public key of authority j together with all public keys appearing previously in the delegation path. As part of its HABS signature the signer proves knowledge of each TBS at every delegation of each attribute that is required to satisfy the policy. Clearly this

delegation process is highly inefficient. Not only does an additional signature need to be verified per delegation (and per attribute), the size of the signature grows linearly in the distance from the root authority. Thus, per attribute, verification of the delegation path is of order $O(k^2)$.

Other Related Work. Attribute-based signatures can be seen as a generalisation of group [11] and ring [34] signatures, in which case identities are viewed as attributes and policies can only contain disjunction over them. The notion of hierarchical delegation in these, more restricted, primitives have been explored in [37] and [29] respectively. Attribute delegation has been widely investigated in anonymous credentials [9,10]. Maji et al. [31] give discussion that ACs are a more powerful primitive than ABS but with efficiency drawbacks, as attribute acquisition typically requires expensive zero-knowledge proofs. Note that in HABS intermediate authorities may know each other, and so as discussed in [12], there is no need to hide their identities from each other during the delegation phase, which in turn helps to omit costly proofs and make this phase more efficient than in the case of ACs. Regardless, we note that ACs with hierarchical delegation have been proposed [4]. Further, a homomorphic ABS scheme [25] has been used to construct non-delegatable anonymous credentials. In this setting, a signer obtains attributes directly from the (multiple) root authorities where combining attributes from different issuers requires an online collaboration. Anonymous Proxy Signatures [17] also allow for verification of anonymous delegation paths back to a root authority. However, tasks that are delegated, when viewed as attributes, remain in the clear and are required for verification of the proxy signature. Homomorphic Signatures [38] have been claimed to be equivalent to Attribute-based Signatures, however this equality has been shown to hold in the weaker security setting that only considers a solitary user. In which, it is impossible to capture the notion of collusion and non-frameability. Finally, we note functional signatures [3,8] also allow for delegation of signing rights. Here, however, keys are dependent on the function f and can only sign on messages that fall within the range. For an attribute-based scheme, this would require keys for each possible combination of attributes a user obtains.

Contribution. We address the suboptimal efficiency of the so-far only (generic) HABS construction [12] and propose a scheme with a completely new delegation mechanism which no longer relies on the consecutive issue of tag-based signatures from higher-level to lower-level authorities on the delegation path. The main novelty in our approach is a smart use of the length-reducing homomorphic trapdoor commitment scheme to multiple group elements from [21] which we extend with delegation capabilities. At a high level, at each delegation the issuing intermediate authority amends the current trapdoor opening such that the existing commitment incorporates the public key of the next-level authority or user to whom the attribute is delegated. With this new delegation mechanism we are able to significantly reduce the lengths of HABS keys and achieve the optimal growth of $O(k|\Psi|)$ for the length of HABS signatures, depending on the length k of the delegation path and size $|\Psi|$ of the signing policy. In particular, verifying delegation of an attribute along the path takes $O(k)$ steps (as opposed

to $O(k^2)$ in [12]). We use the original security model from [12] to show that our construction satisfies the required properties of path anonymity, path traceability, and non-frameability, in the standard model under standard assumptions in bilinear groups and an additional assumption which we justify using the generic group model [36]. Our efficiency improvement claims over [12] are reinforced in a detailed comparison between the two schemes.

2 HABS Model: Entities and Definitions

We start with the description of entities within the HABS ecosystem.

Attribute Authorities. The set of Attribute Authorities (AA) comprises the Root Authority (RA) and Intermediate Authorities (IAs). All AAs can delegate attributes to lower-level IAs and users. The RA is at the top of the hierarchy and upon setup, defines the universe of attributes $\mathbb{A}$. With its key pair (ask_0, apk_0), the RA can delegate a subset of attributes to IAs which hold their own key pairs (ask_i, apk_i), $i > 0$. IAs can further delegate/issue attributes to any end user (aka. signer). In this way a dynamically expandable HABS hierarchy can be established.

Users. Users join the scheme by creating their own key pair (usk, upk), and are issued attributes by possibly multiple AAs.

By Ψ we denote a predicate for some signing policy. A policy-conforming user can use usk to create a HABS signature, provided their issued set of attributes A satisfies the policy, i.e. $\Psi(A') = 1$ for some $A' \subseteq A$. Users are unable to delegate attributes further and thus can be viewed as the lowest tier of the hierarchy. To account for this, when an attribute is delegated to a user a dedicated symbol $\star$ will be used in addition to upk to mark the end of the delegation path.

Warrants. An IA or user, upon joining the HABS scheme, receives a *warrant* **warr** that consists of all their delegated attributes $a \in \mathbb{A}$ and a list of all AAs in each of the delegation paths. Warrants can be updated at any time, i.e. if the owner is issued a new attribute, by appending a new entry with the list of authorities on the delegation path. We use the notation $|\mathbf{warr}|$ to denote the size of the warrant, i.e. the number of attributes stored in the warrant **warr**, and we use $|\mathbf{warr}[a]|$ to denote the length of the delegation path of the attribute $a \in \mathbb{A}$. Upon signing, the user submits a reduced warrant for an attribute set $A' \subseteq A$ that satisfies $\Psi(A') = 1$.

Tracing Authority. The tracing authority (TA) is independent of the hierarchy. Upon receiving a valid HABS signature, it can identify the signer and all authorities on the delegation paths for attributes that the signer used to satisfy the signing policy. The tracing authority can output a publicly verifiable proof $\hat{\pi}$ that the path was identified correctly. The existence of such tracing authority improves the accountability of signers and IAs from possible misbehaviour.

Definition 1 (Hierarchical ABS Scheme [12]). *A scheme* $\texttt{HABS} := (\texttt{Setup}, \texttt{KGen}, \texttt{AttIssue}, \texttt{Sign}, \texttt{Verify}, \texttt{Trace}, \texttt{Judge})$ *consists of the following seven processes:*

- $\texttt{Setup}(1^\lambda)$ *is the initialisation process where based on some security parameter* $\lambda \in \mathbb{N}$*, the public parameters* pp *of the scheme are defined, and the root and tracing authority independently generate their own key pair, i.e. RA's* (ask_0, apk_0) *and TA's* (tsk, tpk)*. In addition, RA defines the universe* $\mathbb{A}$ *of attributes, and a label* $\star$ *for users. We stress that due to dynamic hierarchy, the system can be initialised by publishing (*pp*,* apk_0, tpk*) with* $\mathbb{A}$ *and* $\star$ *contained in* pp*.*
- $\texttt{KGen}(pp)$ *is a key generation algorithm executed independently by intermediate authorities and users. Each entity generates its own key pair, i.e.,* (ask_i, apk_i) *for* $i > 0$ *or* (usk, upk)*.*
- $\texttt{AttIssue}$ $(ask_i, \boldsymbol{warr}_i, A, \{apk_j | upk_j\})$ *is an algorithm that is used to delegate attributes to an authority with* apk_j *or issue them to the user with* upk*. On input of an authority's secret key* ask_i*,* $i \in \mathbb{N}_0$*, its warrant* $\boldsymbol{warr}_i$*, a subset of attributes* A *from* $\boldsymbol{warr}_i$*, and the public key of the entity to which attributes are delegated or issued, it outputs a new warrant for that entity.*
- $\texttt{Sign}$ $((usk, \boldsymbol{warr}), \mathrm{m}, \Psi)$ *is the signing algorithm. On input of the signer's usk and (possibly reduced)* $\boldsymbol{warr}$*, a message* m *and a predicate* Ψ *it outputs a signature* σ*.*
- $\texttt{Verify}$ $(apk_0, (\mathrm{m}, \Psi, \sigma))$ *is a deterministic algorithm that outputs 1 if a candidate signature* σ *on a message* m *is valid with respect to the predicate* Ψ *and 0 otherwise.*
- $\texttt{Trace}$ $(tsk, apk_0, (\mathrm{m}, \Psi, \sigma))$ *is an algorithm executed by the TA on input of its private key* tsk *and outputs either a triple* $(upk, \boldsymbol{warr}, \hat{\pi})$ *if the tracing is successful or* $\perp$ *to indicate its failure. Note that* $\boldsymbol{warr}$ *contains attributes and delegation paths that were used by the signer.*
- $\texttt{Judge}$ $(tpk, apk_0, (\mathrm{m}, \Psi, \sigma),\ (upk, \boldsymbol{warr}, \hat{\pi}))$ *is a deterministic algorithm that checks a candidate triple* $(upk, \boldsymbol{warr}, \hat{\pi})$ *from the tracing algorithm and outputs* 1 *if the triple is valid and* 0 *otherwise.*

A HABS scheme must have the *correctness* property ensuring that any signature σ generated based on an honestly issued warrant will verify and trace correctly. The output $(upk, \mathbf{warr}, \hat{\pi})$ of the tracing algorithm on such signatures will be accepted by the public judging algorithm with overwhelming probability.

2.1 Security Properties

Our security definitions resemble the requirements of *path anonymity*, *path traceability*, and *non-frameability* from [12]. We recall the associated game-based definitions assuming probabilistic polynomial time (PPT) adversaries interacting with HABS entities through the following set of oracles (Fig. 1):

– O_{Reg}: $\mathcal{A}$ registers new IAs and users through this registration oracle, for which a key pair will be generated and added to *List*. The public key is given to

the adversary. Initially, the entity is considered honest, and so the public key is also added to the list HU.

- O_{Corr}: This oracle allows $\mathcal{A}$ to corrupt registered users or IAs. Upon input of a public key, the corresponding private key is given as output if it exists in $List$. The public key is removed from HU so the oracle keeps track of corrupt entities.
- O_{Att}: $\mathcal{A}$ uses this oracle to ask an attribute authority to delegate attributes to either an IA or to the user. In particular, the adversary has control over which attributes are issued and the oracle outputs a warrant **warr** if both parties are registered, otherwise it outputs $\perp$.
- O_{Sig}: $\mathcal{A}$ can ask for a HABS signature from a registered user. The adversary provides the warrant (and implicitly the attributes used), signing policy, message and the public key of the signer. If the attribute set satisfies the policy, and the public key is contained in HU then the signature will be given to $\mathcal{A}$, otherwise $\perp$ is returned.
- O_{Tr}: s $\mathcal{A}$ can ask the TA trace a HABS signature (provided by the adversary) to the output is returned. The TA does verification checks on the signature and upon failure, will return $\perp$.

$O_{\text{Reg}}(\cdot)$ with $(\cdot) = (i)$ and $i \notin HU$

1 : $(sk_i, pk_i) \leftarrow \texttt{KGen}(\text{pp})$
2 : $List \leftarrow List \cup \{(i, pk_i, sk_i)\}$
3 : $HU \leftarrow HU \cup \{i\}$
4 : **return** pk_i

$O_{\text{Corr}}(\cdot)$ with $(\cdot) = (i)$

1 : **if** $i \in HU$ **then**
2 : $HU \leftarrow HU - \{i\}$
3 : **return** sk_i from $List$

$O_{\text{Tr}}(\cdot)$ with $(\cdot) = (\text{m}, \Psi, \sigma)$

1 : **return** $\texttt{Trace}(tsk, apk_0, (\text{m}, \Psi, \sigma))$

$O_{\text{Att}}(\cdot)$

$(\cdot) = (i, \mathbf{warr}_i, a, \{apk_j | upk_j\})$
1 : $L := \{a | (a, pk_a, sk_a) \in List\}$
2 : **if** $i \in L \ \wedge j \in L$ **then**
3 : $\mathbf{warr} \leftarrow \texttt{AttIssue}(ask_i, \mathbf{warr}_i, a, \{apk_j | upk_j\})$
4 : **return warr**
5 : **return** $\perp$

$O_{\text{Sig}}(\cdot)$ with $(\cdot) = (i, \mathbf{warr}, \text{m}, \Psi)$

1 : $A \leftarrow \{a |\ a \in \mathbf{warr}\}$
2 : **if** $i \in HU \ \wedge \ \Psi(A)$ **then**
3 : $\sigma \leftarrow \texttt{Sign}((usk_i, \mathbf{warr}), \text{m}, \Psi)$
4 : **return** σ
5 : **return** $\perp$

Fig. 1. Oracles used in the HABS security experiments.

Path Anonymity. This property extends the anonymity guarantees of traditional ABS schemes and hides not only the identity of the signer but also the identities of all intermediate authorities on delegation paths for attributes

included into the signer's warrant. Path anonymity, as defined in Fig. 2, also ensures that signatures produced by the same signer remain unlinkable. The corresponding experiment requires the adversary to distinguish which warrant and private key were used in the generation of the challenge HABS signature σ_b. In the first phase, $\mathcal{A}_1$ generates a hierarchy of authorities and users, utilising the RA's secret key ask_0. If the warrants created by $\mathcal{A}_1$ are of the same size, a challenge HABS signature σ_b is produced on the randomly chosen user-warrant pair. In the second phase, with access to the tracing oracle, the adversary $\mathcal{A}_2$ must be able to guess the challenge bit b.

Definition 2 (Path Anonymity [12]). *A HABS scheme offers path anonymity if no PPT adversary adv can distinguish between* $\mathbf{Exp}^{\text{pa}-0}_{HABS,\mathcal{A}}$ *and* $\mathbf{Exp}^{\text{pa}-1}_{HABS,\mathcal{A}}$ *defined in Fig. 2, i.e., the following advantage is negligible in* λ*:*

$$\text{Adv}^{\text{pa}}_{HABS,\mathcal{A}}(\lambda) = |\Pr[\text{Exp}^{\text{pa}-0}_{HABS,\mathcal{A}}(\lambda) = 1] - |\Pr[\mathbf{Exp}^{\text{pa}-1}_{HABS,\mathcal{A}}(\lambda) = 1]|.$$

$\mathbf{Exp}^{\text{pa-}b}_{HABS,\mathcal{A}}(\lambda)$

1 : $(\text{pp}, ask_0, tsk) \leftarrow \texttt{Setup}(1^\lambda)$
2 : $(st, (usk_0, \mathbf{warr}_0), (usk_1, \mathbf{warr}_1), m, \Psi) \leftarrow \mathcal{A}_1(\text{pp}, ask_0 : O_{\text{Reg}}, O_{\text{Corr}}, O_{\text{Tr}})$
3 : **if** $|\mathbf{warr}_0| = |\mathbf{warr}_1|$ **then**
4 : $\quad \sigma_0 \leftarrow \texttt{Sign}((usk_0, \mathbf{warr}_0), \text{m}, \Psi),\ \sigma_1 \leftarrow \texttt{Sign}((usk_1, \mathbf{warr}_1), \text{m}, \Psi)$
5 : $\quad$ **if** $\texttt{Verify}(apk_0, (m, \Psi, \sigma_0)) = 1$ and $\texttt{Verify}(apk_0, (m, \Psi, \sigma_1)) = 1$ **then**
6 : $\quad\quad b' \leftarrow \mathcal{A}_2(st, \sigma_b : O_{\text{Tr}})$
7 : $\quad\quad$ **return** $b' \ \wedge$ $\mathcal{A}2$ did not query $O_{\text{Tr}}(tsk, (m, \Psi, \sigma_b))$
8 : **return** 0

Fig. 2. Path-anonymity experiment

Non-frameability. This property, formalised in Fig. 3, captures the notion of unforgeability, i.e., that no PPT adversary can create a HABS signature without having an honestly issued warrant that satisfies the policy, and in particular, they cannot create one on behalf of an user for which the secret key is not known. The adversary wins if either he produces a valid HABS signature, or is able to perform delegation for at least one attribute on behalf of any honest authority anywhere in the delegation path.

Definition 3 (Non-Frameability [12]). *A HABS scheme is non-frameable, if no PPT adversary* $\mathcal{A}$ *can win the experiment* $\mathbf{Exp}^{\text{nf}}_{\text{HABS},\mathcal{A}}$ *defined in Fig. 3, i.e., the following advantage is negligible in* λ*:*

$$\mathbf{Adv}^{\text{nf}}_{HABS,\mathcal{A}}(\lambda) = |\Pr[\mathbf{Exp}^{\text{nf}}_{HABS,\mathcal{A}}(\lambda) = 1]|.$$

$\mathbf{Exp}^{\mathrm{nf}}_{HABS,\mathcal{A}}(\lambda)$

$\mathrm{pp} \leftarrow \mathtt{Setup}(1^\lambda), ask_0 \leftarrow \mathtt{KGen}(1^\lambda), tsk \leftarrow \mathtt{TKGen}(1^\lambda)$
$((\sigma, m, \Psi), (upk_j, \mathbf{warr}, \hat{\pi})) \leftarrow \mathcal{A}(\mathrm{pp}, ask_0, tsk : O_{\mathrm{Att}}, O_{\mathrm{Sig}}, O_{\mathrm{Corr}}, O_{\mathrm{Reg}})$
if $\mathtt{Verify}(apk_0, (\mathrm{m}, \Psi, \sigma)) \ \wedge \ \mathtt{Judge}(tpk, apk_0, (\mathrm{m}, \Psi, \sigma), (upk_j, \mathbf{warr}, \hat{\pi}))$ **then**
  if $j \in HU \wedge \mathcal{A}$ did not query $O_{\mathrm{Sig}}((usk_j, \mathbf{warr}), \mathrm{m}, \Psi)$ **then**, **return** 1
  if $\exists a.\ a \in \mathbf{warr} \implies (apk_0, apk_1, \ldots, apk_n, upk_j, \star) = \mathbf{warr}[a] \wedge$
    $((\exists i \in [0, n-1].\ \mathcal{A}$ did not query $O_{\mathrm{Att}}(i, \cdot, a, apk_{i+1}) \wedge i \in HU) \vee$
    ($\mathcal{A}$ did not query $O_{\mathrm{Att}}(n, \cdot, a, upk_j) \ \wedge n \in HU)$) **then**, **return** 1
return 0

Fig. 3. Non-frameability experiment

$\mathbf{Exp}^{\mathrm{tr}}_{\mathtt{HABS},\mathcal{A}}(\lambda)$

$\mathrm{pp} \leftarrow \mathtt{Setup}(1^\lambda), ask_0 \leftarrow \mathtt{KGen}(1^\lambda), tsk \leftarrow \mathtt{TKGen}(1^\lambda)$
$((\sigma, m, \Psi), (upk, \mathbf{warr}, \hat{\pi})) \leftarrow \mathcal{A}(\mathrm{pp}, tsk : O_{\mathrm{Att}}, O_{\mathrm{Corr}}, O_{\mathrm{Reg}})$
if $\mathtt{Verify}(apk_0, (m, \Psi, \sigma))$ **then**
  if $\mathtt{Trace}(tsk, (m, \Psi, \sigma)) = \perp$ **then**, **return** 1
  if $\mathtt{Judge}(tpk, apk_0, (m, \Psi, \sigma), (upk, \mathbf{warr}, \hat{\pi})) \wedge$
    $(\exists a.\ a \in \mathbf{warr} \implies (apk_0, apk_1, \ldots, apk_n, upk, \star) = \mathbf{warr}[a] \wedge$
      $(\ (\exists i \in [0, n-1].\ i \in HU \wedge (i+1, apk_{i+1}, ask_{i+1}) \notin List) \vee$
        $(n \in HU \wedge (\ \cdot\ , upk, usk) \notin List)\ )\ )$ **then**, **return** 1
return 0

Fig. 4. Path-traceability experiment

Path Traceability. This property, formalised in Fig. 4, ensures that any valid HABS signature can be traced (by the tracing authority) to the signer and the set of authorities that were involved in the issue of attributes used to produce the signature. The adversary is required to output either a HABS signature that verifies but cannot be traced, or one in which the tracing algorithm outputs a warrant for which at least one IA or the user is unknown to the experiment, i.e., were not previously registered in *List*. The attribute-issuing oracle checks that both entities are registered to prevent the trivial attack where adversary asks the oracle to delegate to an unregistered entity.

Definition 4 (Path Traceability [12]). *A HABS scheme offers path traceability if no PPT adversary $\mathcal{A}$ can win the experiment $\mathbf{Exp}^{\mathrm{tr}}_{\mathtt{HABS},\mathcal{A}}$ defined in Fig. 4, i.e., the following advantage is negligible in λ:*

$$\mathbf{Adv}^{\mathrm{tr}}_{HABS,\mathcal{A}}(\lambda) = |\Pr[\mathbf{Exp}^{\mathrm{tr}}_{HABS,\mathcal{A}}(\lambda) = 1]|.$$

3 Our Short HABS Construction

We start with the description of the underlying hardness assumptions and building blocks.

3.1 Underlying Hardness Assumptions

In addition to widely used hardness assumptions in the asymmetric bilinear group setting generated by $\mathcal{BG}(1^\lambda)$, namely q-Srong Diffie-Hellman (q-SDH) [7], Symmetric eXternal Diffie-Hellman (SXDH) [1] and Simultaneous Decision LINear (SDLIN) [26] assumptions, which we do not recall here, our scheme requires the following interactive assumption, which we prove to hold in the generic group model [24,36]. We equip the adversary with an oracle $\mathcal{O}_{X_0,Y_0}$ and assume that it is hard to produce group elements that satisfy the verification equations for an input that has not been queried to the oracle. We note that on input of (X, Y), the oracle can compute each component without knowledge the discrete logarithm of X or Y as it has access to r, s, t_1, t_2.

Assumption 1. *Let* $pp := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathcal{BG}(1^\lambda)$. *The adversary* $\mathcal{A}$ *has access to the oracle* $\mathcal{O}_{X_0,Y_0}(\cdot,\cdot)$ *which on input* (X, Y) *returns* $(A, B, T_1, T_2, T_3, T_4) := (h_2^{x_0 t_1} h_5^{x_0 t_2} h_2^{x t_1} h_5^{y t_1}, h_5^{y_0 t_1} h_4^{y_0 t_2} h_5^{x t_2} h_4^{y t_2}, f_1^{t_1}, f_2^{t_2}, g_2^{t_1}, g_2^{t_2})$ *for* $t_1, t_2 \leftarrow \mathbb{Z}_p^*$. *We assume that for all p.p.t. adversaries* $\mathcal{A}$, *the following probability is negligible in* λ.

$$
Adv_{\mathcal{A}}^{\texttt{Assump1}}(\lambda) := \\
Pr\left[\begin{array}{l}
x_0, y_0, r, s \leftarrow \mathbb{Z}_p^*; h_1 := g_1^s, h_2 := g_1^{s^2}, h_3 := g_1^r, h_4 := g_1^{r^2}, h_5 := g_1^{rs} \\
f_1 := g_2^s, f_2 := g_2^r, X_0 := h_1^{x_0}; Y_0 := h_3^{y_0}; \\
(\hat{A}, \hat{B}, \hat{T}_1, \hat{T}_2, \hat{T}_3, \hat{T}_4, \hat{X}, \hat{Y}) \leftarrow \mathcal{A}^{\mathcal{O}_{X_0,Y_0}}(pp, X_0, Y_0, h_1, h_2, h_3, h_4, h_5, f_1, f_2) : \\
e(\hat{A}, g_2) = e(X_0, \hat{T}_1\hat{T}_2) e(\hat{X}\hat{Y}, \hat{T}_1) \ \wedge \ e(\hat{B}, g_2) = e(Y_0, \hat{T}_1\hat{T}_2) e(\hat{X}\hat{Y}, \hat{T}_2) \\
\wedge \ e(g_1, \hat{T}_1\hat{T}_2) = e(h_1, \hat{T}_3) e(h_3, \hat{T}_4) \quad \text{where } (\hat{X}, \hat{Y}) \text{ was not queried.}
\end{array}\right]
$$

Theorem 1. *Let* $\mathcal{A}$ *denote an adversary in the generic group model against Assumption 1.* $\mathcal{A}$ *has access to oracles for which he makes* q_G *group queries,* q_P *pairing queries, and* q_O *oracle queries. The probability* ϵ *of* $\mathcal{A}$ *winning the game for Assumption 1 is bounded by* $\epsilon \leq 5(q_G + q_P + 6q_O + 11)^2/p$, *where* p *is the prime order of the generic groups.*

Proof. See full version [19].

3.2 Cryptographic Building Blocks

The following building blocks will be used in our HABS construction.

Tag-Based Encryption. A TBE scheme has the same algorithms as a traditional public key encryption scheme, except that its encryption and decryption procedures take an extra tag t as input. A correctly formed TBE ciphertext C will

fail to decrypt if the tag used as input to the decryption algorithm is different from the one used upon encryption. We adopt the TBE scheme from [26] which relies on the SDLIN assumption. It offers selective-tag witness-indistinguishable chosen-ciphertext (st-IND-CCA) security, where an adversary is unable to distinguish between two ciphertexts under the same tag t of their choosing. Kiltz [28] showed that a st-IND-CCA TBE scheme combined with a strongly unforgeable one-time signature, where the one-time verification key is used as the tag, gives rise to an IND-CCA2 secure PKE. Our scheme uses this approach.

One-Time Signature. For the OTS, we use the strongly unforgeable BBS one-time signature scheme from [7]. It consist of three algorithms (KeyGen, Sig, Ver) with a verification key in $\mathbb{G}^2 \times \mathbb{Z}_p^*$ and the corresponding signing key in $\mathbb{Z}_p^{*2}$.

Groth-Sahai Proofs. We use the Groth-Sahai proof system [22] to construct the required non-interactive zero-knowledge proofs NIZK. A GS proof, which consists of five algorithms (Setup, Prove, Verify, SimSetup, SimProve), allows proving relations involving multi-linear, quadratic, and pairing-based equations. We use GS proofs in the asymmetric bilinear group setting with Type-3 curves [18], i.e., where there is no computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$, in which case their security is based on the SXDH assumption [7].

Homomorphic Trapdoor Commitments to Group Elements. The key to our short HABS scheme is the length-reducing homomorphic trapdoor commitment scheme by Groth [21], adopted in our new delegation mechanism. With the HTC scheme, defined by four algorithms (Setup, KeyGen, Commit, Trapdoor), one can use the trapdoor key tk to open a constant-length commitment (c, d) to arbitrary group elements with respect to a commitment key. We observe that due to its construction this HTC scheme has an interesting property that allows a commitment to group elements step-wise, i.e. an opening (a_i, b_i) to elements $g_1, \ldots, g_i$ can be transformed into an opening (a_{i+1}, b_{i+1}) for an extended set of elements $g_1, \ldots, g_{i+1}$ without knowledge of the secret commitment key, i.e., without jeopardising the binding property for the already committed group elements. In our HABS scheme such step-wise extension of an initial commitment (c, d) produced by the root authority allows intermediate authorities, upon delegation, to embed public keys of next-level authorities or users, along with the delegated attribute, by providing appropriate modification to the opening of (c, d), that is without changing its value nor increasing its length. Proving ownership of a delegated attribute amounts to presenting an opening (a, b) for the commitment (c, d) to the attribute and the list of public keys of authorities on the delegation path, i.e., $apk_0, \ldots$,upk, $\star$. In our scheme we use the asymmetric variant of Groth's HTC scheme with security based on the XDLIN assumption [1].

3.3 Specification of Our HABS Scheme

We start with a high-level intuition behind our HABS construction and provide detailed specification in Figs. 5, 6, and 7.

High-Level Overview. As part of the setup process public parameters pp of the scheme are generated. They include the security parameter λ, the description of bilinear groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, the trapdoor key tk for the HTC scheme, the initial 'dummy' HTC commitment (c, d) with an opening (a_0, b_0), and the description of the attribute universe $\mathbb{A}$. The independent tracing authority TA generates the TBE key-pair $(tsk, tpk) := ((\eta_1, \eta_2), (V_1, V_2, V_3, V_4))$ with tpk included into pp. For simplicity we describe the setup phase as a single process involving computations performed by the RA and TA. We stress, however, that generation of $h_1, ..., h_5, f_1, f_2$ must be trusted in that no entity knows the corresponding exponents.

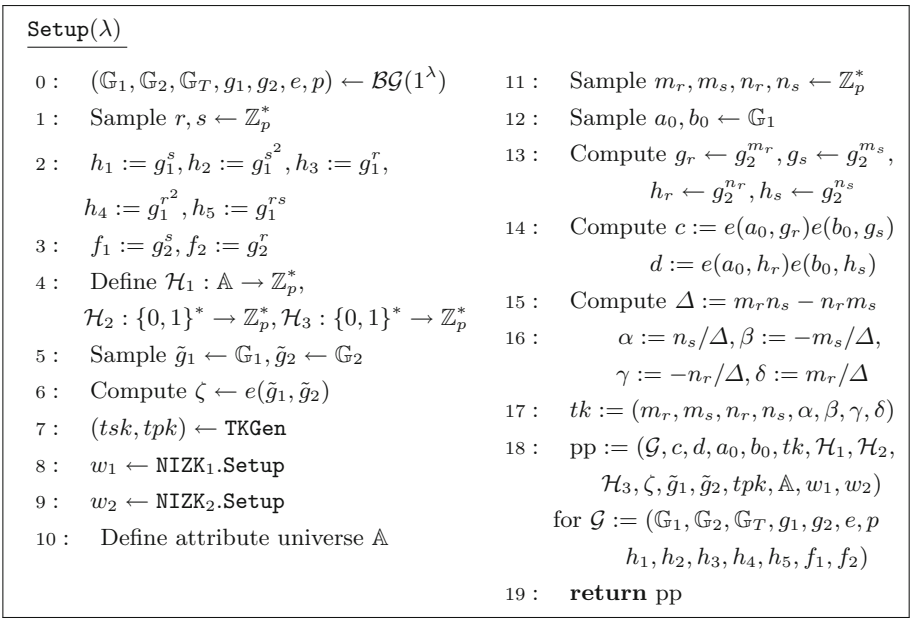

Fig. 5. Setup algorithm of our HABS construction.

In our scheme, all attribute authorities and users generate their own private/public key pairs (ask_i, apk_i) and (usk, upk) respectively, of the form $\{(x, y), (X, Y, Z, \hat{Z})\}$. While only X and Y are used in the verification of attribute delegation which we prove in the signature, the components $(Z, \hat{Z})$ are used in the issuing phase. To ensure an authority creates a delegation that opens to (X, Y), we insist that the validity of a public key is checked prior to delegation. This is done by evaluating $e(XY, h_1) = e(Z, g_2)$ and $e(XY, h_3) = e(\hat{Z}, g_2)$. IAs and users obtain attributes from existing authorities at a higher level in the hierarchy. Ownership of a valid key-pair (ask_i, apk_i) allows authorities to delegate attributes further down the hierarchy and to the users, by manipulating the opening of the initial commitment (c, d).

With the trapdoor key tk an authority can create an opening (a_i, b_i) for (c, d) to the path that includes delegate's public key e.g. apk_j. Rather than opening directly, the issuer first creates randomisation tokens $T_1, T_2, T_3, T_4 \in \mathbb{G}_2$ and opens to these instead. It then uses T_1 and T_2 as *one-time* commitment keys to open to apk_j and the delegated attribute att, that is hashed into the message space using $g^{\mathcal{H}_1(att)}$. The randomisation tokens T_1 and T_2 are used to prevent forgeries (where the adversary combines multiple openings and in doing so, forges an opening to a new public key) whereas T_3 and T_4 are used to verify the well-formedness of T_1 and T_2, by evaluating $e(g_1, T_1T_2) = e(h_1, T_4)e(h_3, T_3)$. The issuing authority updates the (possibly empty) warrant with opening (a_i, b_i), his public key apk_i and the randomisation tokens (T_1, T_2, T_3, T_4). As tk is part of public parameters, any IA in the hierarchy is able to perform the delegation procedure, where it receives (a_{i-1}, b_{i-1}) from its issue and generates (a_i, b_i) for the next delegation. When delegating to users, an issuing IA will open (c, d) to a designated element $\star \in \mathbb{G}$, in addition to the user's public key upk and the attribute att. A warrant contains the trapdoor opening and a list of all public keys of AAs that appear in the delegation path for any issued attribute.

Upon signing, the user first generates an `OTS` key-pair $(otssk, otsvk) := \{(k_1, k_2), (K_1, K_2, \kappa)\}$ and an opening to $\mathcal{H}_3(otsvk)$ by modifying the opening (a_0, b_0) using his public key upk and the trapdoor key tk. The reduced **warr** along with upk are encrypted in a `TBE` ciphertext under the TA's public key tpk and tag $\mathcal{H}_3(otsvk)$. The signing policy Ψ is modelled as a monotone span program, with labelling function ρ that maps rows from $\mathbf{S}$ to the

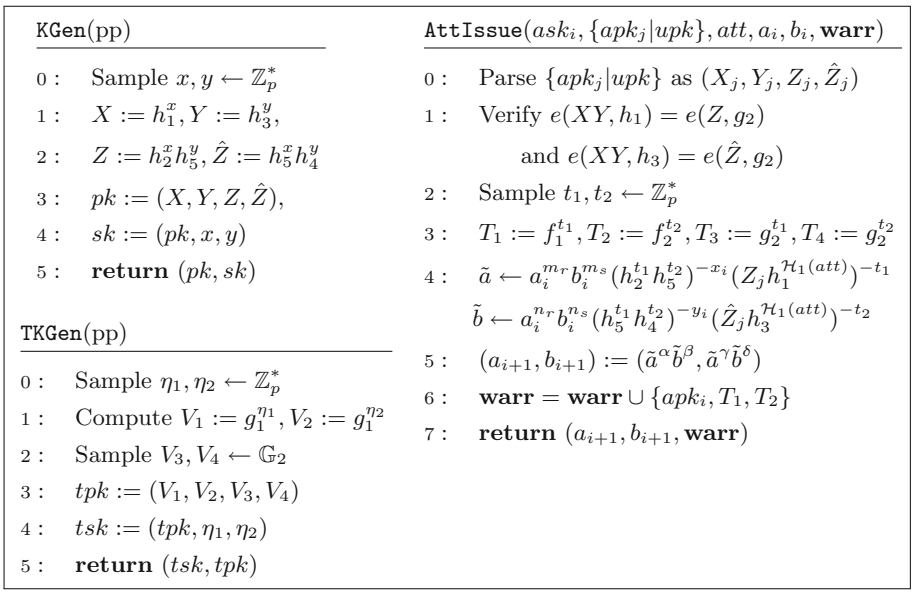

Fig. 6. Key generation and issue of attributes in our `HABS` construction.

attribute set $\mathbb{A}$. The signer proves that this set satisfies Ψ by computing a vector $\mathbf{z}$ such that $\mathbf{zS} = [1, 0, ..., 0]$, where any non-zero entry $\mathbf{z}_i$ implies $\rho(i) \in \mathbf{warr}$. A NIZK proof π is then computed using Groth-Sahai framework with witness $(upk, \mathbf{warr}, \mathbf{z}, \tilde{r}, \tilde{s})$ for the following relation:

$$
\begin{aligned}
&\big((a,b), \mathbf{warr}, upk, \mathbf{z}\big), \big(\Psi, otsvk, apk_0, C, tpk\big) : \mathbf{z}S = [1,0,...,0] \\
&\wedge\ (\forall i.\ z_i \neq 0 \implies att_i = \rho(i)\ \wedge\ (apk_{i_1}, ..., apk_{i_n}, apk_{i_{n+1}} := upk) \in C \\
&\wedge\ c^{|\mathbf{warr}|+1} = e(a, g_r)e(b, g_s)e(X, g_2^{\mathcal{H}_3(otsvk)}) \\
&\qquad\qquad \Pi_i \Pi_{n=0}^{k} e(X_{i_n}, T_{1,i_n} T_{2,i_n}) e(X_{i_{n+1}} Y_{i_{n+1}} g_1^{\mathcal{H}_1(att)}, T_{1,i_n}) \\
&\wedge\ d^{|\mathbf{warr}|+1} = e(a, h_r)e(b, h_s)e(Y, g_2^{\mathcal{H}_3(otsvk)}) \\
&\qquad\qquad \Pi_i \Pi_{n=0}^{k} e(Y_{i_n}, T_{1,i_n} T_{2,i_n}) e(X_{i_{n+1}} Y_{i_{n+1}} g_1^{\mathcal{H}_1(att)}, T_{2,i_n}) \\
&\qquad\qquad \wedge\ e(g_1, \Pi_i \Pi_{n=0}^{k} T_{1,i_n} T_{2,i_n}) = e(h_1, \Pi_i \Pi_{n=0}^{n} T_{4,i_n}) e(h_3, \Pi_i \Pi_{n=0}^{k} T_{3,i_n}).
\end{aligned}
$$

Sign$(usk, m, \Psi, \{att_j, a_j, b_j, \mathbf{warr}_j\}_{j \in J})$

0 : $(k_1, k_2, k_3) \leftarrow \mathbb{Z}_p^*$

1 : $otsvk := (\tilde{g}_2^{k_1}, \tilde{g}_2^{k_2}, k_3)$

2 : Compute $\mathbf{z}$ s.t. $\mathbf{z}S = [1, 0, ..., 0]$

3 : $T_1 := f_1^{t_1}, T_2 := f_2^{t_2}$,
$T_3 := g_2^{t_1}, T_4 := g_2^{t_2}$

4 : $a' \leftarrow a_i^{m_r} b_i^{m_s} (h_2^{t_1} h_5^{t_2})^{-x} h_1^{-t_1 \mathcal{H}_3(otsvk)}$
$b' \leftarrow a_i^{n_r} b_i^{n_s} (h_5^{t_1} h_4^{t_2})^{-y} h_3^{-t_2 \mathcal{H}_3(otsvk)}$

5 : $(a', b') := (\tilde{a}^{\alpha} \tilde{b}^{\beta}, \tilde{a}^{\gamma} \tilde{b}^{\delta})$

6 : $(a, b) = (a' \cdot \Pi a_j, b' \cdot \Pi b_j)$

7 : $C \leftarrow$ TBE.Enc$(tpk, \mathbf{warr}, upk, a, b, \{\mathcal{H}_1(otsvk)\})$

8 : $\pi \leftarrow$ NIZK$_1$.Prove$((upk, \mathbf{z}, \mathbf{warr}, a, b) : (C, otsvk, tpk, apk_0, \Psi) \in \mathcal{R})$

9 : $H \leftarrow \mathcal{H}_2(\pi || C || \Psi || m || otsvk)$

10 : $\sigma_o \leftarrow \tilde{g}_1^{1/(k_1 + H + k_2 k_3)}$

11 : **return** $(\sigma_o, C, \pi, otsvk)$

Verify(pk, σ, m, Ψ)

0 : Parse σ as $(ots, \pi, otsvk)$

1 : $H \leftarrow \mathcal{H}_2(\pi || C || \Psi || m || otsvk)$

2 : **return** NIZK.Verify(π)
$\wedge\ e(\sigma_o, \tilde{g}_2^{k_1} \cdot \tilde{g}_2^{H} \cdot \tilde{g}_2^{k_2 k_3}) = \zeta$

Trace(tsk, σ, m, Ψ)

0 : **if** Verify$(\sigma, \mathrm{m}, \Psi) = 1$ **then**

1 : **warr** $\leftarrow$ TBE.Dec(tsk, C, t) for $t = \mathcal{H}_1(otsvk)$

2 : $\hat{\pi} \leftarrow$ NIZK$_2$.Prove$(tsk : (otsvk, C, tpk, (apk_0, \mathbf{warr})))$

3 : **return** $(\mathbf{warr}, \hat{\pi})$

Judge$(tpk, \mathbf{warr}, \sigma)$

0 : Verify$(ask_0, (\sigma, \mathrm{m}, \Psi))$
$\wedge$ NIZK$_2$.Verify(π_2)

Fig. 7. Sign, Verify, Trace and Judge algorithms of our HABS construction.

The message m and policy Ψ are then bound to this proof and ciphertext by hashing $\mathcal{H}_2(\pi, \mathrm{C}, \Psi, m)$, before an OTS signature σ_o is produced with $otssk$. The resulting signature is verified with respect to the public parameters of the scheme, and the RA's public key apk_0 by verifying the OTS signature and the NIZK proof.

As part of the tracing procedure, executed by TA with knowledge of tsk, the ciphertext C is decrypted to obtain the warrant **warr**, signer's public key upk,

and the opening (a, b). A publicly verifiable NIZK proof $\hat{\pi}$ is created with witness tsk for the statement ($otsvk$, C, tpk, (apk_0, **warr**)) and relation:

$$\texttt{TBE.Dec}(tsk, C, \mathcal{H}_3(otsvk)) = (upk, \mathbf{warr}, a, b).$$

We give a detailed construction for the Groth-Sahai proofs $\texttt{NIZK}_1$ and $\texttt{NIZK}_2$ in Appendix D.

3.4 Security Analysis

In this section we prove that our construction meets HABS security properties of path anonymity, non-frameability and path traceability.

Lemma 1. *The* HABS *construction from Figs. 5, 6 and 7 offers path anonymity, if SXDH, SDLIN and q-SDH hold in* $\mathcal{G}$.

Proof. We follow a game-based approach and show that the advantage of the PPT adversary $\mathcal{A}$ in the path-anonymity experiment for the HABS construction from Figs. 5, 6 and 7, is bounded by the advantages of the constructed adversaries for the underlying primitives. We assume that adversary $\mathcal{A}$ asks n user registration queries and the probability for sampling one of these users is $1/n$.

Game G_0**:** This game is defined to the be the experiment $\mathbf{Exp}^{\text{pa-}b}_{\texttt{HABS},\mathcal{A}}(\lambda)$ in Fig. 2, where the 2-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is required to distinguish between the HABS signatures $\sigma_0 = (\sigma^0_o, \mathrm{C}_0, \pi_0, otsvk_0)$ and $\sigma_1 = (\sigma^1_o, \mathrm{C}_1, \pi_1, otsvk_1)$.
Game G_1**:** We define the game G_1 as G_0 where the check "$\mathcal{A}_2$ did not query $O_{\mathrm{Tr}}(m, \Psi, \sigma_b)$" is enforced by the O_{Tr} oracle available to $\mathcal{A}_2$, which aborts the game if this is the case. The probability from G_0 to G_1 is preserved.
Game G_2**:** The game G_2 is obtained from G_1 where, on the output of O_{Tr}, we replace the $\texttt{NIZK}_2$ proof $\hat{\pi}$ with $\hat{\pi}'$ from the simulator $\texttt{NIZK}_2\texttt{.SimProve}$. We also replace Setup by SimSetup for $\texttt{NIZK}_2$. This prevents the case where $\mathcal{A}$ may "extract" tsk from $\texttt{NIZK}_2$ proofs. Thus, for all future O_{Tr} oracle calls we use the simulated $\texttt{NIZK}_2$ proof. The probability that $\mathcal{A}$ can distinguish between these two games is bounded by the advantage of the zero-knowledge adversary $\mathcal{B}_{nizk2}$ for $\texttt{NIZK}_2$. For our instantiation of GS proofs, this is reduced to the SXDH assumption [22].
Game G_3**:** Let G_3 be the game obtained from G_2 where we replace the proof π_b from the challenge signature $\sigma_b = (\sigma_{o,b}, C_b, \pi_b, otsvk_b)$ with the simulated proof π'_b by calling $\texttt{NIZK}_1\texttt{.Sim}$ on $(\mathrm{C}_b, otsvk_b, tpk, apk_0, \Psi)$. Additionally, we replace $\texttt{NIZK}_1\texttt{.Setup}$ by $\texttt{NIZK}_1\texttt{.SimSetup}$. The probability that $\mathcal{A}$ can distinguish between games G_2 and G_3 is bounded by the advantage of the zero-knowledge adversary $\mathcal{B}_{nizk1}$ for $\texttt{NIZK}_1$ proof. Similarly, this property is implied by SXDH.
Game G_4**:** Game G_4 only differs from game G_3 in that we abort if $\mathcal{A}_2$ queries $O_{\mathrm{Tr}}(m, \Psi, (\sigma_o, \mathrm{C}_b, \pi, otsvk_b))$. The adversary $\mathcal{A}$ is only able to distinguish between these games if it can produce a valid OTS signature σ_o for the message (C_b, π, m, Ψ) and public key $otsvk_b$, without knowledge of the secret key $otssk_b$. Thus, the capabilities of the adversary $\mathcal{A}$ to distinguish between these

two games is bounded by the advantage of the adversary $\mathcal{B}_{ots}$ against the strong unforgeability of the OTS scheme, which is reduced to the q-SDH assumption [7].
Game G_5: Game G_5 is defined to be G_4, except we additionally do a check for any queries $\mathcal{A}_2$ makes do not contain the challenge ciphertext, that is $O_{\mathrm{Tr}}(m, \Psi, (\sigma_o, \mathrm{C}_b, \pi, otsvk))$. If so the game is aborted. The output of O_{Tr} is for G_4 and G_5 is the same, as the oracle returns $\perp$ if the tag $otsvk_b$ for C is different from $otsvk$ received as input. Hence, the probability is preserved.
Game G_6: The game G_6 is the same as G_5, except that we move the OTS key generation from the signature generation phase into the setup of the experiment. Note that only one key pair needs to be created in this game since the adversary only sees the challenge signature. This step is necessary to utilise the st-IND-CCA property of the TBE scheme. The probability is unchanged from game G_5 to G_6.
Game G_7: Let G_7 be the game obtained from G_6 where the TBE ciphertext C_b from the challenge signature $\sigma_b = (\sigma_o^b, C_b, \pi_b', otsvk_b)$ is replaced with the C_0. The adversary $\mathcal{A}$ is unable to query a ciphertext $C' \neq \mathrm{C}_b$ for the same tag $\mathcal{H}_3(otsvk)$ as a result of game G_4. Further, any query to the oracle for a tag $t' \neq \mathcal{H}_3(otsvk)$ will also fail as decryption of C_b is dependent on the correct tag. Thus, the ability of the adversary $\mathcal{A}_2$ to distinguish between the ciphertexts C_0 and C_b is bounded by the advantage of the st-IND-CCA adversary $\mathcal{B}_{ind}$. For out instantiation, this property of TBE is implied by SDLIN [26].

The experiment G_7 provides $\mathcal{A}$ with the same challenge signature independent of b that $\mathcal{A}$ is asked to guess. Additionally, due to the zero-knowledge property of $\mathtt{NIZK}_2$ used in G_1, $\mathcal{A}$ does not have access to tsk. Therefore, the probability that the adversary wins game G_7 is $1/2$ and hence the advantage of $\mathcal{A}$ to win this experiment is 0. □

Lemma 2. *The* HABS *construction from Figs. 5, 6 and 7 is non-frameable, if $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ are second-preimage resistant hash functions, and q-SDH, SXDH and Assumption 1 hold in $\mathcal{G}$.*

Proof. We begin by first splitting the non-frameability experiment from Fig. 3 into two experiments based on the winning condition of the adversary $\mathcal{A}$. The first $\mathbf{Exp}_1$, defined in Fig. 8, captures the probability of the adversary $\mathcal{A}$ to create a forged HABS signature. The second experiment $\mathbf{Exp}_2$ is the same as $\mathbf{Exp}_1$ except that the event "$j \in HU \ \wedge$ $\mathcal{A}$ did not query $O_{\mathrm{Sig}}((usk_j, \mathbf{warr}), \Psi, \mathrm{m})$" is replaced with

$$\begin{aligned}
&\text{“}\exists a.\ a \in \mathbf{warr} \implies (apk_0, apk_1, \ldots, apk_n, upk_j, \star) = \mathbf{warr}[a] \wedge \\
&\quad (\ (\exists 0 \leq i \leq n-1.\ \mathcal{A} \text{ did not call } O_{\mathrm{Att}}(i, \cdot, a, apk_{i+1}) \wedge i \in HU) \vee \\
&\quad\ (\mathcal{A} \text{ did not call } O_{\mathrm{Att}}(n, \cdot, a, upk_u) \ \wedge n \in HU))\text{”}
\end{aligned}$$

We capture the probability of winning the non-frameability experiment by the probability that $\mathcal{A}$ wins either $\mathbf{Exp}_1$ or $\mathbf{Exp}_2$.

We first bound the advantage of the adversary for the experiment $\mathbf{Exp}_1$. Intuitively, we consider the output of the adversary and argue that each component must coincide with a call to the signing oracle. The forgery is denoted by

$\mathbf{Exp}_1$ - The $\mathbf{Exp}^{\mathrm{nf}}_{\mathtt{HABS},\mathcal{A}}(\lambda)$ where $\mathcal{A}$ did not query $O_{\mathrm{Sig}}((usk, \mathbf{warr}), \Psi, \mathrm{m})$

$(\mathrm{pp}, ask_0, tsk) \leftarrow \mathtt{Setup}(1^\lambda)$
$((m, \Psi, \sigma), (upk_j, \mathbf{warr}, (\pi, \hat{\sigma}_s))) \leftarrow \mathcal{A}(\mathrm{pp}, ask_0, tsk : O_{\mathrm{Att}}, O_{\mathrm{Sig}}, O_{\mathrm{Corr}}, O_{\mathrm{Reg}})$
$(\sigma_o, \mathrm{C}, \pi, otsvk) = \sigma$
if $\mathtt{NIZK}_1.\mathtt{Verify}((\mathrm{C}, otsvk, tpk, apk_0, \Psi), \pi) \wedge$
  $\mathtt{OTS}.\mathtt{Verify}(otsvk, (m, \Psi, \mathrm{C}, \pi), \sigma_o) \wedge$
  $\mathtt{NIZK}_2.\mathtt{Verify}(tpk, (otsvk, \mathrm{C}, upk_j, \mathbf{warr}, \sigma_s), \hat{\pi}) \wedge$
  $j \in HU \wedge$
  $\mathcal{A}$ did not query $O_{\mathrm{Sig}}((usk_j, \mathbf{warr}), \Psi, \mathrm{m})$ **then**
  return 1
return 0

Fig. 8. Experiment $\mathbf{Exp}_1$

$(upk', \mathbf{warr}', m', \Psi'), (\sigma'_o, \mathrm{C}', \pi', otsvk')$. We take each element of the tuple $(upk_j, \mathbf{warr}, m, \Psi)$ and try to reason about its relation with their prime counterpart.

Given n calls to the registration oracle, we model the probability the adversary can guess which oracle constructs the keys for a particular user uniformly, i.e. is equal to $1/n$.

Game G_0**.** This game is defined as $\mathbf{Exp}_1$ where the query restriction "$\mathcal{A}$ did not query $O_{\mathrm{Sig}}((usk_j, \mathbf{warr}), m, \Psi)$" in instead enforced by a membership check $(upk_u, \mathbf{warr}, m, \Psi) \notin SigL$ for the list $SigL$. We also introduce the list $SigLO$ that stores the input and output of the O_{Sig} oracle. Both lists are initialised empty at the beginning of the experiment, and are updated with the inputs, and additionally, the outputs of the O_{Sig} oracle, respectively. The probability is preserved between $\mathbf{Exp}_1$ and G_0.

Game G_1**.** This game is defined exactly as G_0 with the exception of an additional check that the opening (a, b) for (c, d) contains the path $e(X_i, g^{\mathcal{H}_3(otsvk)})$ and $e(Y_i, g^{\mathcal{H}_3(otsvk)})$, respectively. The success probability of a soundness adversary for $\mathtt{NIZK}_1$ bounds the distinguishability of G_1 from G_0. That is, $\mathcal{A}$ can only distinguish between these two games if it is able to generate a valid $\mathtt{NIZK}_1$ proof for a false statement, namely that (a, b) does not open to $g^{\mathcal{H}_3(otsvk)}$. Soundness for our instantiation of $\mathtt{NIZK}_1$ is implied by the SXDH assumption [22].

Game G_2**.** The game G_2 is obtained from G_1 by adding the condition $(upk_j, \star, \star, \star) \notin SigL$. The adversary in G_2 managed to create a valid opening (a, b) for upk_u to $g^{\mathcal{H}_3(otsvk)}$, without having access to the user's secret key usk_j (since $j \in HU$). The capabilities of $\mathcal{A}$ in this experiment are upper-bounded by the advantage of an adversary against Assumption 1 and the second-preimage property of $\mathcal{H}_3$.

Game G_3**.** Game G_3 is defined to be Game G_1, but where $\mathcal{A}$ made at least one signing query that contains user upk_j. Therefore, there exists $((upk_j, \mathbf{warr}', m', \Psi'), (\sigma'_o, \mathrm{C}', \pi', otsvk')) \in SigL$ with $(\mathbf{warr}, m, \Psi) \neq (\mathbf{warr}', m', \Psi')$ as $(upk_j,$

$\mathbf{warr}, m, \Psi) \notin SigL$ but $(upk_j, \mathbf{warr}', m', \Psi') \in SigL$. The probability is preserved between G_1 and G_3.

Game G_4**.** We define G_4 as the game G_3 where $otsvk \neq otsvk'$. In this case, the adversary $\mathcal{A}$ is able to provide a forged opening to $g^{\mathcal{H}(otsvk')}$ without knowledge of usk_j. This is similar to the method of computing the bound for G_2, except that now $\mathcal{A}$ asks signature queries for upk. It is also bounded by an adversary against Assumption 1.

Game G_5**.** We define game G_5 as the game G_3 where $(m, \Psi) \neq (m', \Psi')$. At this point, we have $otsvk = otsvk'$ and $upk = upk_j$ for some j. Thus, if $\mathcal{A}$ can distinguish between G_5 and G_3 then it is able to provide a forgery for the OTS scheme by signing a message that contains (m', Ψ') without knowledge of $otssk$, or break the second preimage property of $\mathcal{H}_2$.

Game G_6**.** We define game G_6 as the game G_5 where $(m, \Psi) = (m', \Psi')$. Because of the $(\mathbf{warr}, m, \Psi) \neq (\mathbf{warr}', m', \Psi')$ restriction, we have $\mathbf{warr}' \neq \mathbf{warr}$. The correctness property of the encryption scheme TBE that builds C' now implies $\mathrm{C} \neq \mathrm{C}'$ under the tag $t = \mathcal{H}_3(otsvk)$. The probability that the adversary can distinguish between G_6 and G_5 is upper-bounded by an adversary $\mathcal{B}'_{cor}$ against the correctness of TBE, which is implied by the SDLIN assumption [26].

Game G_7**.** Let G_7 is the same as G_6 but with $\mathrm{C} \neq \mathrm{C}'$. Assuming second-preimage resistance of $\mathcal{H}_2$, the adversary $\mathcal{A}$ managed to create a forged OTS signature without knowledge of $otssk$. Therefore, the probability of success for adversary $\mathcal{A}$ in this game is bounded by the advantage of an OTS-forger $\mathcal{B}'_{ots}$. The q-SDH assumption implies the BBS signature is strongly unforgeable [7].

From the sequence of games $G_0, \ldots, G_7$, it follows that the probability of $\mathbf{Exp}_1$ is bounded by the unforgeability of OTS, zero-knowledge of $\mathtt{NIZK}_1$, correctness of TBE, and computational hardness of Assumption 1.

The experiment $\mathbf{Exp}_2$ captures the case where the adversary $\mathcal{A}$ is able to provide a forged delegation for an honest authority apk_i and some attribute att. In this case, $\mathcal{A}$ is bounded by the hardness of Assumption 1 and the second preimage property of $\mathcal{H}_1$.

Lemma 3. *The* HABS *construction from Figs. 5, 6 and 7 offers path traceability, if SXDH, SDLIN and Assumption 1 hold in* $\mathcal{G}$*.*

Proof. See full version [19].

Theorem 2. *The HABS scheme in Figs. 5, 6 and 7 offers path-anonymity, non-frameability and path-traceability if* $\mathcal{H}_1$*,* $\mathcal{H}_2$ *and* $\mathcal{H}_3$ *are second-preimage resistant hash functions and SXDH, SDLIN, q-SDH and Assumption 1 hold in* $\mathcal{G}$*.*

Proof. The result follows from Lemmas 1, 2 and 3. □

4 Efficiency Comparison

We first compare the warrant sizes for our scheme and [12]. For a single attribute, an authority at level 1 (with respect to the root authority at level 0) has a warrant

size of 6 group elements (of the form $\mathbb{G}_1^2 \times \mathbb{G}_2^4$). Further delegation to level 2 adds a further 6 elements in $\mathbb{G}_1^2 \times \mathbb{G}_2^4$. A delegation from the root authority contains the opening (a, b) (as part of the 6 elements) which is updated by subsequent delegations, however the warrant must now also contain the issuers public key (X_i, Y_i) in $\mathbb{G}_1^2$. This generalises, and for a user at level k, the size of the warrant is $6k$ for a single attribute. Likewise, if we extend the number of attributes in the warrant to $|A|$, each of which has a delegation path of length k, then the warrant has $6k|A|$ group elements.

In contrast, for a single attribute issued to a level-1 entity, the warrant in [12] contains $7\lceil\frac{12+2m}{m-2}\rceil$ group elements, where m is the size of the message space used in the TBS instantiation. A level-2 delegation increases this to $7\lceil\frac{24+4m}{m-2}\rceil + 2m + 12$ elements, and this generalises for a single attribute that is issued to a level k entity to $7\lceil\frac{k(12+2m)}{m-2}\rceil + (k-1)(2m+12)$ elements. Similarly, a warrant that contains $|A|$ attributes adds a linear factor of $|A|$ to this term. To give a concrete comparison, a user with 3 attributes at level 4 of the hierarchy would have a warrant containing 72 group elements in our scheme, as opposed to 208 elements for an optimal choice of m (i.e., $m = 10$) in the scheme from [12]. Since m would be chosen in advance during the setup phase, the warrant would unlikely reach its optimal bound and for any suboptimal choice of m, the warrant grows linearly in this parameter.

Next, in Table 1 we compare the sizes of public keys (of users and authorities) and the lengths of signatures generated by our scheme and [12]. By β we denote the size of the span program representing the policy Ψ. As before k is the maximum length of a delegation path, $|\Psi|$ is the number of attributes needed to satisfy the signing policy, and m is the size of the message space for the TBS scheme used in [12].

Table 1. Comparison of key and signature sizes.

		Dragan et al. [12]		This Work		
		$\mathbb{G}$	$\mathbb{Z}_p$	$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{Z}_p$
Public Keys	upk	14	-	4	-	-
	apk	$12+2m$	-	4	-	-
Sig.	ots	3	1	2	1	1
	C	$\frac{5(6+m)k(k-1)\lvert\Psi\rvert}{(m-2)} + 110$	-	$6(2k-1)\lvert\Psi\rvert + 12$	$4(2k-1)\lvert\Psi\rvert + 8$	-
	π	$\frac{28(6+m)k(k-1)\lvert\Psi\rvert}{(m-2)} + 18$	2β	8	$2k\lvert\Psi\rvert + 8$	β

In addition to being more efficient and shorter than [12], our scheme, in fact, produces HABS signatures of *optimal* length, from the asymptotic point of view. The need to provide path traceability, where the TA must be able to reveal the entire delegation path along with delegated attributes from a valid HABS signature implies the $O(k|\Psi|)$ growth of its length. This means that in order to reduce this bound path-tracability property would need to be relaxed.

Finally, our scheme brings a few other efficiency improvements. The use of Type-3 pairings results in fewer group elements and the possibility to achieve the same level security for a smaller choice of the prime p [18], which would give

rise to smaller groups and faster operations than in the symmetric setting. In addition, we can adopt batch verification techniques available for Groth-Sahai proofs [6] to speed up the computations.

5 Conclusion

We proposed a direct construction of Hierarchical Attribute-based Signatures (HABS) with a new delegation process based on length-reducing homomorphic trapdoor commitments. Our HABS scheme significantly reduces the lengths of warrants, public keys and signatures in comparison to the so-far only known (generic) HABS construction. Moreover, due to the need to support the path-traceability requirement, our HABS scheme achieves optimal signature length growth of $O(k|\Psi|)$ for delegations paths of size k and signing policies of size $|\Psi|$. Our technique of step-wise embedding of new group elements into the homomorphic trapdoor commitment can be considered to be of independent interest, e.g., it could add support for delegation to other privacy-preserving signature schemes that rely on homomorphic trapdoor commitments, e.g. [2].

Acknowledgements. Daniel Gardham was supported by the UK Government PhD studentship scheme. Mark Manulis was supported by the EPSRC project TAPESTRY (EP/N02799X). The authors thank the anonymous reviewers of ACNS 2019 for their valuable comments.

References

1. Abe, M., Chase, M., David, B., Kohlweiss, M., Nishimaki, R., Ohkubo, M.: Constant-size structure-preserving signatures: generic constructions and simple assumptions. J. Cryptol. **29**, 833–878 (2016)
2. Abe, M., Haralambiev, K., Ohkubo, M.: Signing on elements in bilinear groups for modular protocol design. IACR Cryptology ePrint Archive, p. 133 (2010)
3. Backes, M., Meiser, S., Schröder, D.: Delegatable functional signatures. In: Cheng, C.-M., Chung, K.-M., Persiano, G., Yang, B.-Y. (eds.) PKC 2016. LNCS, vol. 9614, pp. 357–386. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49384-7_14
4. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_7
5. Bellare, M., Fuchsbauer, G.: Policy-based signatures. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 520–537. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_30
6. Blazy, O., Fuchsbauer, G., Izabachène, M., Jambert, A., Sibert, H., Vergnaud, D.: Batch Groth–Sahai. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 218–235. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13708-2_14
7. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_4

8. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_29
9. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Persiano, G., Galdi, C. (eds.) SCN 2002. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36413-7_20
10. Chaum, D.: Security without identification: transaction systems to make big brother obsolete. Commun. ACM **28**(10), 1030–1044 (1985)
11. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_22
12. Drăgan, C.-C., Gardham, D., Manulis, M.: Hierarchical attribute-based signatures. In: Camenisch, J., Papadimitratos, P. (eds.) CANS 2018. LNCS, vol. 11124, pp. 213–234. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00434-7_11
13. El Kaafarani, A., Ghadafi, E.: Attribute-based signatures with user-controlled linkability without random oracles. In: O'Neill, M. (ed.) IMACC 2017. LNCS, vol. 10655, pp. 161–184. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71045-7_9
14. El Kaafarani, A., Ghadafi, E., Khader, D.: Decentralized traceable attribute-based signatures. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 327–348. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04852-9_17
15. El Kaafarani, A., Katsumata, S.: Attribute-based signatures for unbounded circuits in the ROM and efficient instantiations from lattices. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10770, pp. 89–119. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76581-5_4
16. Escala, A., Herranz, J., Morillo, P.: Revocable attribute-based signatures with adaptive security in the standard model. In: Nitaj, A., Pointcheval, D. (eds.) AFRICACRYPT 2011. LNCS, vol. 6737, pp. 224–241. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21969-6_14
17. Fuchsbauer, G., Pointcheval, D.: Anonymous proxy signatures. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 201–217. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85855-3_14
18. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Discrete Appl. Math. **156**(16), 3113–3121 (2008)
19. Gardham, D., Manulis, M.: Hierarchical attribute-based signatures: short keys and optimal signature length. Cryptology ePrint Archive, Report 2019/382 (2019). https://eprint.iacr.org/2019/382
20. Ghadafi, E.: Stronger security notions for decentralized traceable attribute-based signatures and more efficient constructions. In: Nyberg, K. (ed.) CT-RSA 2015. LNCS, vol. 9048, pp. 391–409. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16715-2_21
21. Groth, J.: Homomorphic Trapdoor Commitments to Group Elements. Cryptology ePrint Archive, Report 2009/007 (2009)
22. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
23. Herranz, J.: Attribute-based signatures from RSA. TCS **527**, 73–82 (2014)
24. Jager, T., Rupp, A.: The semi-generic group model and applications to pairing-based cryptography. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 539–556. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_31

25. Kaaniche, N., Laurent, M., Rocher, P.-O., Kiennert, C., Garcia-Alfaro, J.: PCS, a privacy-preserving certification scheme. In: Data Privacy Management, Cryptocurrencies and Blockchain Technology, pp. 239–256 (2017)
26. Kakvi, S.A.: Efficient fully anonymous group signatures based on the Groth group signature scheme. Master's thesis, University College London (2010)
27. Khader, D., Chen, L., Davenport, J.H.: Certificate-free attribute authentication. In: Parker, M.G. (ed.) IMACC 2009. LNCS, vol. 5921, pp. 301–325. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10868-6_18
28. Kiltz, E.: Chosen-ciphertext security from tag-based encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_30
29. Krzywiecki, Ł., Sulkowska, M., Zagórski, F.: Hierarchical ring signatures revisited – unconditionally and perfectly anonymous schnorr version. In: Chakraborty, R.S., Schwabe, P., Solworth, J. (eds.) SPACE 2015. LNCS, vol. 9354, pp. 329–346. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24126-5_19
30. Li, J., Au, M.H., Susilo, W., Xie, D., Ren, K.: Attribute-based signature and its applications. In: ACM ASIACCS 2010, pp. 60–69. ACM (2010)
31. Maji, H.K., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 376–392. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_24
32. Okamoto, T., Takashima, K.: Decentralized attribute-based signatures. In: Kurosawa, K., Hanaoka, G. (eds.) PKC 2013. LNCS, vol. 7778, pp. 125–142. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36362-7_9
33. Okamoto, T., Takashima, K.: Efficient attribute-based signatures for non-monotone predicates in the standard model. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 35–52. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_3
34. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32
35. Sakai, Y.: Practical attribute-based signature schemes for circuits from bilinear map. IET Inf. Secur. **12**, 184–193 (2018)
36. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_18
37. Trolin, M., Wikström, D.: Hierarchical group signatures. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 446–458. Springer, Heidelberg (2005). https://doi.org/10.1007/11523468_37
38. Tsabary, R.: An equivalence between attribute-based signatures and homomorphic signatures, and new constructions for both. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10678, pp. 489–518. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_16