Short Attribute-Based Signatures for Threshold Predicates

Javier Herranz, Fabien Laguillaumie, Benoît Libert, Carla Ràfols

Short Attribute-Based Signatures for Threshold Predicates

Javier Herranz[1], Fabien Laguillaumie[2], Benoît Libert[3], and Carla Ràfols[1]

[1] Universitat Politècnica de Catalunya, Dept. Matemàtica Aplicada IV (Spain)
[2] GREYC, Université de Caen Basse-Normandie (France)
[3] Université catholique de Louvain, ICTEAM Institute – Crypto Group (Belgium)

Abstract. Attribute-based cryptography is a natural solution for fine-grained access control with respect to security policies. In the case of attribute-based signatures (ABS), users obtain from an authority their secret keys as a function of the attributes they hold, with which they can later sign messages for any predicate satisfied by their attributes. A verifier will be convinced of the fact that the signer's attributes satisfy the signing predicate while remaining completely ignorant of the identity of the signer. In many scenarios where authentication and anonymity are required, like distributed access control mechanisms in ad hoc networks, the bandwidth is a crucial and sensitive concern. The signature size of all previous ABS schemes grows linearly in the number of attributes involved in the signing predicate. We propose the first two attribute-based signature schemes with constant size signatures. Their security is proven in the selective-predicate and adaptive-message setting, in the standard model, under chosen message attacks, with respect to some algorithmic assumptions related to bilinear groups. The described schemes are for the case of threshold predicates, but they can be extended to admit some other (more expressive) kinds of monotone predicates.

Keywords. Attribute-based signatures, constant signature size, efficiency.

1 Introduction

Attribute-based cryptography offers a real alternative to public-key cryptography when the systems to be protected also require anonymity among users following a security policy. In this setting, users obtain their secret keys from an authority as a function of their attributes. The operation involving the secret key proves somehow that the user holds a certain subset of attributes, without leaking information on his identity or on his total set of attributes.

One of the major issues in attribute-based cryptography is to save bandwidth, and in particular to get ciphertexts or signatures of constant size, *i.e.,* not depending on the number of involved attributes. Other important issues are the construction of systems achieving security in the strongest possible model and being as expressive as possible, *i.e.,* admitting a wide variety of policies. The goal of this work is to address the first question in the context of signature design.

Attribute-based cryptography first appeared in [16] with an attribute-based encryption scheme, as an extension of fuzzy identity-based cryptosystems [29]. Since then, the notion of attribute-based encryption (ABE for short, conjugated into *key policy* or *ciphertext policy*) has received a lot of attention (see for example [2, 18, 20]), with many attempts to reduce the length of the ciphertexts (see [13, 18, 1]).

Attribute-based signatures (shortened as ABS in the sequel) have been introduced more recently in [24] (see also [30, 21, 22]). They are related to the notion of (threshold) ring signatures [28, 7] or mesh signatures [6], but offer much more flexibility and versatility to design secure complex systems, since the signatures are linked not to the users themselves, but to their attributes. As a consequence, these signatures have a wide range of applications, like private access control, anonymous credentials, trust negotiations, distributed access control mechanisms for ad hoc networks, attribute-based messaging... (see [24] for detailed descriptions of applications). In terms of security, ABS must first satisfy unforgeability, which guarantees that a signature cannot be computed

by a user who does not have the right attributes, even if he colludes with other users by pooling together their secret keys. The other required security feature is the privacy of user's attributes, in the sense that a signature should not leak any information about the actual attributes that have been employed to produce it.

Related work. Among the schemes proposed up to now, those of Maji, Prabhakaran, Rosulek proposed in [24] work for very expressive signing predicates, but their most practical scheme is only proven secure in the generic group model. In [27], this scheme is claimed to be "almost optimally efficient", although its signatures' length grows linearly in the size of the span program (which is greater than the number of involved attributes in the signing predicate). Our result shows that this claim is not true, at least for some families of predicates (*e.g.*, threshold). Some other instantiations in [24] are secure in the standard model, but are pretty inefficient (*i.e.,* the signature size is linear in the security parameter of the scheme), because they use Groth-Sahai proofs for relations between the bits of elements in the group. Okamoto and Takashima designed in [27] a *fully* secure ABS in the standard model which supports general non-monotone predicates. The scheme is not built upon non-interactive zero-knowledge proof systems, but on dual pairing vector spaces [26] and uses proof techniques from functional encryption [20]. Escala, Herranz and Morillo also proposed in [14] a fully secure ABS in the standard model, with the additional property of revocability, meaning that a third party can extract the identity of a signer in case of dispute (thanks to a secret that can be computed by the master entity). *None* of the previous schemes achieves constant-size signatures.

Our contribution. In this paper we propose the first attribute-based signature schemes which produce such short signatures, and which are proven secure in the selective-predicate setting (*i.e.,* not *fully* secure), in the standard model. We design two constant-size ABS schemes, both built (non-generically) on two different constant-size attribute-based encryption schemes. In both schemes, let n denote the maximum size of the admitted signing predicates.

- Our first scheme supports (weighted) threshold predicates for small[4] universes of attributes. Its design is inspired by the constant-size ciphertext-policy ABE scheme from [18] by Herranz, Laguillaumie and Ràfols, in the sense that the signer implicitly proves his ability to decrypt a ciphertext by using the Groth-Sahai proof systems [17], and by binding the signed message (and the corresponding predicate) to the signature using a technique suggested by Malkin, Teranishi, Vahlis and Yung [23]. The signature consists of 15 group elements, and the secret key of a user holding a set Ω of attributes is made up with $|\Omega| + n$ elements. Our scheme is selective-predicate and adaptive-message unforgeable under chosen message attacks if the augmented multi-sequence of exponents computational Diffie-Hellman assumption [18] and the Decision Linear assumption [5] hold. The privacy of the attributes involved in the generation of a signature is preserved, under the Decision Linear assumption.
- Our second scheme supports threshold predicates (as well as compartmented and hierarchical predicates) for *large* universes of attributes. It is built upon Attrapadung, Libert and de Panafieu's key-policy ABE scheme from [1] and has signatures consisting of *only* 3 group elements. The secret keys are longer than in the first scheme, since they include $(2n+2) \times (|\Omega|+n)$ group elements. The assumption underlying the selective-predicate and adaptive-message unforgeability under chosen message attacks is the more classical n-Diffie-Hellman exponent assumption, and the scheme protects the privacy of the involved attributes unconditionally.

[4] which means that the number of possible attributes is polynomial in the security parameter, which is highly sufficient for most applications

Organization of the paper. We give the algorithmic setting and define the syntactics of attribute-based signatures and the required security properties in Section 2. In Section 3 we describe our first scheme, for threshold signing predicates, and prove its security. We do the same for our second scheme in Section 4, and discuss possible extensions of both schemes to more general signing predicates in Section 5. Concluding remarks are given in Section 6.

2 Background

2.1 Notation

We will treat a vector as a column vector, unless stated otherwise. For any vector $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)^\top \in \mathbb{Z}_p^n$, and any element g of a group $\mathbb{G}$, $g^{\vec{\alpha}}$ stands for the vector of group elements $(g^{\alpha_1}, \ldots, g^{\alpha_n})^\top \in \mathbb{G}^n$. For $\vec{a}, \vec{z} \in \mathbb{Z}_p^n$, we denote their inner product as $\langle \vec{a}, \vec{z} \rangle = \vec{a}^\top \vec{z} = \sum_{i=1}^n a_i z_i$. Given $g^{\vec{a}}$ and $\vec{z}$, $(g^{\vec{a}})^{\vec{z}} := g^{\langle \vec{a}, \vec{z} \rangle}$ is computable without knowing $\vec{a}$. For equal-dimension vectors $\vec{A}$ and $\vec{B}$ containing exponents or group elements, $\vec{A} \cdot \vec{B}$ stands for their component-wise product. When $\vec{C} = (C_1, C_2, C_3)^\top \in \mathbb{G}^3$ is a vector of group elements and if $g \in \mathbb{G}$, we denote by $E(g, \vec{C})$ the vector of pairing values $\big(e(g, C_1), e(g, C_2), e(g, C_3)\big)^\top$. We denote by I_n the identity matrix of size n. For any set U, we define $2^U = \{S \mid S \subseteq U\}$. Additionally, given a set $S \subset \mathbb{Z}_p$, and some $i \in S$, the ith Lagrange basis polynomial is $\Delta_i^S(X) = \prod_{j \in S \setminus \{i\}} (X - j)/(i - j)$.

2.2 Complexity Assumptions

Our two schemes work in the setting of bilinear groups. That is, we use a pair of multiplicative groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p with an efficiently computable mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ s.t. $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G} \times \mathbb{G}$, $a, b \in \mathbb{Z}$ and $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$.

The security of our first scheme is partially based on the hardness of the computational version of a problem appeared in [18] under the name of *augmented multi-sequence of exponents decisional Diffie-Hellman problem*. Its decisional version was proven to be hard in generic groups.

Definition 1 ($(\tilde{\ell}, \tilde{m}, \tilde{t})$-aMSE-CDH - [18]). *The $(\tilde{\ell}, \tilde{m}, \tilde{t})$-augmented multi-sequence of exponents computational Diffie-Hellman ($(\tilde{\ell}, \tilde{m}, \tilde{t})$-aMSE-CDH) problem related to the group pair $(\mathbb{G}, \mathbb{G}_T)$ is to compute $T = e(g_0, h_0)^{\kappa \cdot f(\gamma)}$ on input: the vector $\vec{x}_{\tilde{\ell}+\tilde{m}} = (x_1, \ldots, x_{\tilde{\ell}+\tilde{m}})^\top$, whose components are pairwise distinct elements of $\mathbb{Z}_p$ which define the polynomials*

$$f(X) = \prod_{i=1}^{\tilde{\ell}} (X + x_i) \text{ and } g(X) = \prod_{i=\tilde{\ell}+1}^{\tilde{\ell}+\tilde{m}} (X + x_i),$$

and the values

$$\begin{cases} g_0, g_0^{\gamma}, \ldots, g_0^{\gamma^{\tilde{\ell}+\tilde{t}-2}}, & g_0^{\kappa \cdot \gamma \cdot f(\gamma)}, & (l.1) \\ g_0^{\omega\gamma}, \ldots, g_0^{\omega\gamma^{\tilde{\ell}+\tilde{t}-2}}, & & (l.2) \\ g_0^{\alpha}, g_0^{\alpha\gamma}, \ldots, g_0^{\alpha\gamma^{\tilde{\ell}+\tilde{t}}}, & & (l.3) \\ h_0, h_0^{\gamma}, \ldots, h_0^{\gamma^{\tilde{m}-2}}, & h_0^{\kappa \cdot g(\gamma)} & (l.4) \\ h_0^{\omega}, h_0^{\omega\gamma}, \ldots, h_0^{\omega\gamma^{\tilde{m}-1}}, & & (l.5) \\ h_0^{\alpha}, h_0^{\alpha\gamma}, \ldots, h_0^{\alpha\gamma^{2(\tilde{m}-\tilde{t})+3}} & & (l.6) \end{cases}$$

where $\kappa, \alpha, \gamma, \omega$ are unknown random elements of $\mathbb{Z}_p$ and g_0 and h_0 are generators of $\mathbb{G}$.

The other (decisional) hard problem that we need for the security analysis of our first signature scheme is the Decision Linear Problem.

Definition 2 (DLIN - [5]). *In a group $\mathbb{G}$ of prime order p, the* Decision Linear Problem *(DLIN) is to distinguish the distributions $(g, g^a, g^b, g^{a\cdot\delta_1}, g^{b\cdot\delta_2}, g^{\delta_1+\delta_2})$ and $(g, g^a, g^b, g^{a\cdot\delta_1}, g^{b\cdot\delta_2}, g^{\delta_3})$, with $a, b, \delta_1, \delta_2, \delta_3 \xleftarrow{R} \mathbb{Z}_p$.*

This problem is to decide if vectors $\vec{g_1} = (g^a, 1, g)^\top$, $\vec{g_2} = (1, g^b, g)^\top$ and $\vec{g_3} = (g^{a\delta_1}, g^{b\delta_2}, g^{\delta_3})^\top$ are linearly dependent in the $\mathbb{Z}_p$-module $\mathbb{G}^3$ formed by entry-wise multiplication.

Finally, the security of our second attribute-based signature scheme is based on the hardness of the n-Diffie-Hellman Exponent problem.

Definition 3 (n-DHE - [8]). *In a group $\mathbb{G}$ of prime order p, the n-*Diffie-Hellman Exponent *(n-DHE) problem is, given a tuple $(g, g^\gamma, g^{\gamma^2}, \ldots, g^{\gamma^n}, g^{\gamma^{n+2}}, \ldots, g^{\gamma^{2n}})$ where $\gamma \xleftarrow{R} \mathbb{Z}_p$, $g \xleftarrow{R} \mathbb{G}$, to compute $g^{\gamma^{n+1}}$.*

The generic hardness of this problem is a consequence of a general result given in [4]. In addition, the assumption that this problem is hard is non-interactive and thus falsifiable [25].

2.3 Groth-Sahai Proof Systems

To simplify the description, our first scheme uses Groth-Sahai proofs based on the DLIN assumption, although instantiations based on the symmetric external Diffie-Hellman assumption are also possible. In the DLIN setting, the Groth-Sahai proof systems [17] use a common reference string comprising vectors $\vec{g_1}, \vec{g_2}, \vec{g_3} \in \mathbb{G}^3$, where $\vec{g_1} = (g_1, 1, g)^\top$, $\vec{g_2} = (1, g_2, g)^\top$ for some $g_1, g_2, g \in \mathbb{G}$. To commit to $X \in \mathbb{G}$, one sets $\vec{C} = (1, 1, X)^\top \cdot \vec{g_1}^r \cdot \vec{g_2}^s \cdot \vec{g_3}^t$ with $r, s, t \xleftarrow{R} \mathbb{Z}_p$. When proofs should be perfectly sound, $\vec{g_3}$ is set as $\vec{g_3} = \vec{g_1}^{\xi_1} \cdot \vec{g_2}^{\xi_2}$ with $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p^*$. Commitments $\vec{C} = (g_1^{r+\xi_1 t}, g_2^{s+\xi_2 t}, X \cdot g^{r+s+t(\xi_1+\xi_2)})^\top$ are then Boneh-Boyen-Shacham (BBS) ciphertexts [5] that can be decrypted using $a = \log_g(g_1)$, $b = \log_g(g_2)$.

In the witness indistinguishability (WI) setting, defining $\vec{g_3} = \vec{g_1}^{\xi_1} \cdot \vec{g_2}^{\xi_2} \cdot (1, 1, g^{-1})^\top$ gives linearly independent $\{\vec{g_1}, \vec{g_2}, \vec{g_3}\}$ and $\vec{C}$ is a perfectly hiding commitment. Under the DLIN assumption, the two settings are indistinguishable.

To prove that committed group elements satisfy certain relations, the Groth-Sahai techniques require one commitment per variable and one proof element (made of a constant number of group elements) per relation. Such proofs are available for pairing-product relations, which are of the type

$$\prod_{i=1}^{n} e(\mathcal{A}_i, \mathcal{X}_i) \cdot \prod_{i=1}^{n} \cdot \prod_{j=1}^{n} e(\mathcal{X}_i, \mathcal{X}_j)^{a_{ij}} = t_T,$$

for variables $\mathcal{X}_1, \ldots, \mathcal{X}_n \in \mathbb{G}$ and constants $t_T \in \mathbb{G}_T$, $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{G}$, $a_{ij} \in \mathbb{Z}_p$, for $i, j \in \{1, \ldots, n\}$.

At some additional cost (typically, auxiliary variables have to be introduced), pairing-product equations admit non-interactive zero-knowledge (NIZK) proofs (this is the case when the target element t_T has the special form $t_T = \prod_{i=1}^{t} e(S_i, T_i)$, for constants $\{(S_i, T_i)\}_{i=1}^{t}$ and some $t \in \mathbb{N}$): on a simulated common reference string (CRS), prepared for the WI setting, a trapdoor makes it possible to simulate proofs without knowing the witnesses.

As far as efficiency goes, linear pairing product equations (where $a_{ij} = 0$ for all i, j) consist of only 3 group elements and we only need linear equations here.

2.4 Syntax of Threshold Attribute-Based Signatures

Since we will focus on *threshold* signing predicates, we describe the syntax and security model of attribute-based signatures with respect to these threshold predicates $\Gamma = (t, S)$: the sender chooses a subset S of the universe of attributes and a threshold t such that $1 \leq t \leq |S|$, and signs a message Msg for the pair (t, S). A verifier will be convinced that the signature comes from a user who holds t or more attributes in S. The algorithms and security model for ABS schemes supporting more general signing predicates can be described in a very similar way.

An *attribute-based signature scheme* $\mathsf{ABS} = (\mathsf{ABS.TSetup}, \mathsf{ABS.MSetup}, \mathsf{ABS.Keygen}, \mathsf{ABS.Sign}, \mathsf{ABS.Verify})$ consists of five probabilistic polynomial-time (PPT, for short) algorithms. In the context of threshold predicates, their specification is the following:

- $\mathsf{TSetup}(\lambda, \mathcal{P}, n)$: is the randomized *trusted setup* algorithm taking as input a security parameter λ, an attribute universe $\mathcal{P}$ and an integer $n \in \mathsf{poly}(\lambda)$ which is an upper bound on the size of threshold policies. It outputs a set of public parameters pms (which contains λ, $\mathcal{P}$ and n). An execution of this algorithm is denoted as $\mathsf{pms} \leftarrow \mathsf{ABS.TSetup}(1^\lambda, \mathcal{P}, n)$.
- $\mathsf{MSetup}(\mathsf{pms})$: is the randomized *master setup* algorithm, that takes as input pms and outputs a master secret key msk and the corresponding master public key mpk. We write $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{ABS.MSetup}(\mathsf{pms})$ to denote an execution of this algorithm.
- $\mathsf{Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$: is a (possibly randomized) *key extraction* algorithm which takes as input the public parameters pms, the master keys mpk and msk, and a set of attributes $\Omega \subset \mathcal{P}$ held by the requesting user. The output is a private key SK_Ω. We refer to an execution of this protocol as $SK_\Omega \leftarrow \mathsf{ABS.Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$.
- $\mathsf{Sign}(\mathsf{pms}, \mathsf{mpk}, SK_\Omega, \mathsf{Msg}, \Gamma)$: is a randomized *signing* algorithm which takes as input the public parameters pms, the master public key mpk, a secret key SK_Ω, a message Msg and a threshold signing policy $\Gamma = (t, S)$ where $S \subset \mathcal{P}$ and $1 \leq t \leq |S| \leq n$. It outputs a signature σ. We denote the action taken by the signing algorithm as $\sigma \leftarrow \mathsf{ABS.Sign}(\mathsf{pms}, \mathsf{mpk}, SK_\Omega, \mathsf{Msg}, \Gamma)$.
- $\mathsf{Verify}(\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}, \sigma, \Gamma)$: is a deterministic *verification* algorithm taking as input the public parameters pms, a master public key mpk, a message Msg, a signature σ and a threshold predicate $\Gamma = (t, S)$. It outputs 1 if the signature is deemed valid and 0 otherwise. We write $b \leftarrow \mathsf{ABS.Verify}(\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}, \sigma, \Gamma)$ to refer to an execution of the verification protocol.

For correctness, if $\Gamma = (t, S)$, it is required that

$$\mathsf{ABS.Verify}(\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}, \mathsf{ABS.Sign}(\mathsf{pms}, \mathsf{mpk}, SK_\Omega, \mathsf{Msg}, \Gamma), \Gamma) \;=\; 1,$$

whenever $|\Omega \cap S| \geq t$ and the values $\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, SK_\Omega$ have been obtained by properly executing the algorithms $\mathsf{ABS.TSetup}$, $\mathsf{ABS.MSetup}$ and $\mathsf{ABS.Keygen}$.

2.5 Security of Threshold Attribute-Based Signatures

Unforgeability and privacy are the typical requirements for attribute-based signature schemes.

Unforgeability. An attribute-based signature scheme must satisfy the usual property of unforgeability, even against a group of colluding users that put their secret keys together. In this work we consider a relaxed notion where the attacker *selects* the signing policy $\Gamma^\star = (t^\star, S^\star)$ that he wants to attack at the beginning of the game. Note however that the message $\mathsf{Msg}^\star$ whose signature is eventually forged is not selected in advance. The attacker can ask for valid signatures for messages and signing policies of his adaptive choice. The resulting property of *selective-predicate and adaptive-message unforgeability under chosen message attacks* ($\mathsf{sP\text{-}UF\text{-}CMA}$, for short) is defined by considering the following game that an attacker $\mathcal{F}$ plays against a challenger:

Definition 4. *Let λ be an integer. Consider the following game between a probabilistic polynomial time (PPT) adversary $\mathcal{F}$ and its challenger.*

Initialization. *The challenger begins by specifying a universe of attributes $\mathcal{P}$ as well as an integer $n \in \mathsf{poly}(\lambda)$, which are sent to $\mathcal{F}$. Then, $\mathcal{F}$ selects a subset $S^\star \subset \mathcal{P}$ of attributes such that $|S^\star| \leq n$ and a threshold $t^\star \in \{1, \ldots, |S^\star|\}$. These define a threshold predicate $\Gamma^\star = (t^\star, S^\star)$.*

Setup. *After receiving $\Gamma^\star$ from $\mathcal{F}$, the challenger runs $\mathsf{pms} \leftarrow \mathsf{ABS.TSetup}(1^\lambda, \mathcal{P}, n)$ and $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{ABS.MSetup}(\mathsf{pms})$, and sends $\mathsf{pms}, \mathsf{mpk}$ to the forger $\mathcal{F}$.*

Queries. *$\mathcal{F}$ can interleave private key and signature queries.*

- ***Private key queries.*** *$\mathcal{F}$ adaptively chooses a subset of attributes $\Omega \subset \mathcal{P}$ under the restriction that $|\Omega \cap S^\star| < t^\star$ and must receive $SK_\Omega \leftarrow \mathsf{ABS.Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$ as the answer.*
- ***Signature queries.*** *$\mathcal{F}$ adaptively chooses a pair (Msg, Γ) consisting of a message Msg and a threshold predicate $\Gamma = (t, S)$ such that $1 \leq t \leq |S| \leq n$. The challenger chooses an arbitrary attribute set $\Omega \subset \mathcal{P}$ such that $|\Omega \cap S| \geq t$, runs $SK_\Omega \leftarrow \mathsf{ABS.Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$ and computes a signature $\sigma \leftarrow \mathsf{ABS.Sign}(\mathsf{pms}, \mathsf{mpk}, SK_\Omega, \mathsf{Msg}, \Gamma)$ which is returned to $\mathcal{F}$.*

Forgery. *At the end of the game, $\mathcal{F}$ outputs a pair $(\mathsf{Msg}^\star, \sigma^\star)$. We say that $\mathcal{F}$ is successful if:*

- *$\mathsf{ABS.Verify}(\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}^\star, \sigma^\star, \Gamma^\star) = 1$, and*
- *$\mathcal{F}$ has not made any signature query for the pair $(\mathsf{Msg}^\star, \Gamma^\star)$.*

The forger' success in breaking the sP-UF-CMA security of the ABS scheme is defined as

$$\mathsf{Succ}^{\mathsf{sP\text{-}UF\text{-}CMA}}_{\mathcal{F},\mathsf{ABS}}(\lambda) = \Pr[\mathcal{F} \text{ wins}].$$

A threshold attribute-based signature scheme ABS is selective-predicate adaptive-message unforgeable (sP-UF-CMA unforgeable) *if $\mathsf{Succ}^{\mathsf{sP\text{-}UF\text{-}CMA}}_{\mathcal{F},\mathsf{ABS}}(\lambda)$ is negligible with respect to the security parameter λ, for any polynomial time adversary $\mathcal{F}$.*

Privacy (of Involved Attributes). This property ensures that the only information that an attribute-based signature leaks about the actual attributes that have been used to produce it is the fact that they satisfy the specified signing predicate. Privacy must hold even against attackers that control the master entity and is defined *via* the following game between an adversary $\mathcal{D}$ and its challenger. Depending on the computational resources allowed to $\mathcal{D}$ and on its success probability, we can define computational privacy and perfect (unconditional) privacy.

Definition 5. *Let λ be an integer, and consider the following game between a distinguisher $\mathcal{D}$ and its challenger.*

Setup. *The adversary $\mathcal{D}$ specifies a universe of attributes $\mathcal{P}$ and an integer $n \in \mathsf{poly}(\lambda)$, that are sent to the challenger. The challenger runs $\mathsf{pms} \leftarrow \mathsf{ABS.TSetup}(1^\lambda, \mathcal{P}, n)$ and sends pms to $\mathcal{D}$. The adversary $\mathcal{D}$ runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{ABS.MSetup}(\mathsf{pms})$ and sends $(\mathsf{mpk}, \mathsf{msk})$ to the challenger (who must verify consistency of this master key pair).*

Challenge. *$\mathcal{D}$ outputs a tuple $(\Gamma, \Omega_0, \Omega_1, \mathsf{Msg})$, where $\Gamma = (t, S)$ is a threshold predicate such that $1 \leq t \leq |S| \leq n$ and Ω_0, Ω_1 are attribute sets satisfying $|\Omega_b \cap S| \geq t$ for each $b \in \{0, 1\}$. The challenger picks a random bit $\beta \xleftarrow{R} \{0, 1\}$, runs $SK_{\Omega_\beta} \leftarrow \mathsf{ABS.Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega_\beta)$ and computes $\sigma^\star \leftarrow \mathsf{ABS.Sign}(\mathsf{pms}, \mathsf{mpk}, SK_{\Omega_\beta}, \mathsf{Msg}, \Gamma)$, which is sent as a challenge to $\mathcal{A}$.*

Guess. *$\mathcal{D}$ outputs a bit $\beta' \in \{0, 1\}$ and wins if $\beta' = \beta$.*

The advantage of $\mathcal{D}$ is measured in the usual way, as the distance $\mathsf{Adv}^{\mathsf{Priv}}_{\mathcal{D},\mathsf{ABS}}(\lambda) := |\Pr[\beta' = \beta] - \frac{1}{2}|$.

A threshold attribute-based signature scheme ABS *is said to be* computationally private *if* $\mathsf{Adv}^{\mathit{Priv}}_{\mathcal{D},\mathsf{ABS}}(\lambda)$ *is negligible with respect to the security parameter* λ*, for any distinguisher* $\mathcal{D}$ *running in polynomial time and is said to be* perfectly / unconditionally private *if* $\mathsf{Adv}^{\mathit{Priv}}_{\mathcal{D},\mathsf{ABS}}(\lambda) = 0$*, for any distinguisher* $\mathcal{D}$ *(with possibly unbounded computational power).*

3 A First Short Attribute-Based Signature Scheme for Threshold Predicates

We present here our first scheme to produce attribute-based signatures with constant size, for threshold predicates. The secret key sk_Ω for a user holding a set of attributes Ω contains $|\Omega| + n$ elements, where n is the maximum size of the attribute set for any signing policy. This construction is for "small" universes of attributes $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_\eta\}$, for some integer $\eta \in \mathbb{N}$, as public parameters have linear size in η; therefore, η must be polynomial in the security parameter of the scheme. Attributes $\{\mathsf{at}_i\}_{i=1}^{\eta}$ are arbitrary strings which some encoding function ς maps to $\mathbb{Z}_p^*$. Since the scheme is a small universe construction, we may set $n = \eta$ in the description hereafter.

The construction builds on the ABE scheme put forth by Herranz *et al.* [18]. The intuition is to have the signer implicitly prove his ability to decrypt a ciphertext corresponding to that ABE scheme. This non-interactive proof is generated using the Groth-Sahai proof systems [17], by binding the signed message (and the corresponding predicate) to the non-interactive proof using a technique suggested by Malkin *et al.* [23]. In some sense, this technique can be seen as realizing signatures of knowledge in the standard model: it consists in embedding the message to be signed in the Groth-Sahai CRS by calculating part of the latter as a "hash value" of the message. As noted in [23], Waters' hash function [32] is well-suited to this purpose since, in the security proof, it makes it possible to answer signing queries using simulated NIZK proofs. At the same time, with non-negligible probability, adversarially-generated signatures are produced using a perfectly sound Groth-Sahai CRS and they thus constitute real proofs, from which witnesses can be extracted.

In [23], the above technique was applied to an instantiation of Groth-Sahai proofs based on the Symmetric eXternal Diffie-Hellman assumption (and thus asymmetric pairings). In this section, we adapt this technique so as to get it to work with symmetric pairings and the linear assumption.

▶ $\mathsf{TSetup}(\lambda, \mathcal{P}, n)$: the trusted setup algorithm conducts the following steps.

1. Choose groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$ with an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Select generators $g, h \xleftarrow{R} \mathbb{G}$ and also choose a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^k$, for some $k \in \mathsf{poly}(\lambda)$.
2. Define a suitable injective encoding ς sending each of the n attributes $\mathsf{at} \in \mathcal{P}$ onto an element $\varsigma(\mathsf{at}) = x \in \mathbb{Z}_p^\star$. Choose a set $\mathcal{D} = \{d_1, \ldots, d_{n-1}\}$ consisting of $n-1$ pairwise different elements of $\mathbb{Z}_p^*$, which must also be different from the encoding of any attribute in $\mathcal{P}$. For any integer i lower or equal to $n-1$, we denote as $\mathcal{D}_i$ the set $\{d_1, \ldots, d_i\}$.
3. Generate Groth-Sahai reference strings by choosing random generators $g_1, g_2 \xleftarrow{R} \mathbb{G}$ and defining vectors $\vec{g_1} = (g_1, 1, g)^\top \in \mathbb{G}^3$ and $\vec{g_2} = (1, g_2, g)^\top \in \mathbb{G}^3$. Then, for each $i \in \{0, \ldots, k\}$, pick $\xi_{i,1}, \xi_{i,2} \xleftarrow{R} \mathbb{Z}_p$ at random and define a vector $\vec{g_{3,i}} = \vec{g_1}^{\xi_{i,1}} \cdot \vec{g_2}^{\xi_{i,2}} = \left(g_1^{\xi_{i,1}}, g_2^{\xi_{i,2}}, g^{\xi_{i,1}+\xi_{i,2}}\right)^\top$. Exponents $\{(\xi_{i,1}, \xi_{i,2})\}_{i=0}^k$ can then be discarded as they are no longer needed.

The resulting public parameters are

$$\mathsf{pms} = \Big(\mathcal{P},\ n,\ \lambda,\ \mathbb{G},\ \mathbb{G}_T,\ g,\ h,\ \vec{g_1},\ \vec{g_2},\ \{\vec{g_{3,i}}\}_{i=0}^k,\ H,\ \varsigma,\ \mathcal{D}\Big).$$

▶ $\mathsf{MSetup}(\mathsf{pms})$: the master setup algorithm picks at random $\alpha, \gamma \in \mathbb{Z}_p^*$ and sets $u = g^{\alpha\gamma}$ and $v = e(g^\alpha, h)$.

The master secret key is $\mathsf{msk} = (\alpha, \gamma)$ and the master public key is

$$\mathsf{mpk} = \Big(u,\ v,\ g^{\alpha},\ \Big\{h^{\alpha\gamma^i}\Big\}_{i=0,\ldots,2n-1} \Big).$$

▶ $\mathsf{Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$: to generate a key for the attribute set Ω, pick $r \xleftarrow{R} \mathbb{Z}_p^*$ and compute the private key

$$SK_{\Omega} = \Big(\Big\{g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}}\Big\}_{\mathsf{at}\in\Omega}, \Big\{h^{r\gamma^i}\Big\}_{i=0,\ldots,n-2}, h^{\frac{r-1}{\gamma}} \Big).$$

▶ $\mathsf{Sign}(\mathsf{pms}, \mathsf{mpk}, SK_{\Omega}, \mathsf{Msg}, \Gamma)$: to sign $\mathsf{Msg} \in \{0,1\}^*$ w.r.t. the policy $\Gamma = (t, S)$, where $S \subset \mathcal{P}$ is an attribute set of size $s = |S| \leq n$ and $1 \leq t \leq s \leq n$, the algorithm returns $\perp$ if $|\Omega \cap S| < t$. Otherwise, it first parses SK_{Ω} as

$$SK_{\Omega} = \Big(\Big\{g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}}\Big\}_{\mathsf{at}\in\Omega}, \Big\{h^{r\gamma^i}\Big\}_{i=0,\ldots,n-2}, h^{\frac{r-1}{\gamma}} \Big)$$

and conducts the following steps.

1. Let Ω_S be any subset of $\Omega \cap S$ with $|\Omega_S| = t$. From all $\mathsf{at} \in \Omega_S$, compute the value

$$A_1 = \mathsf{Aggregate}(\{g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}}, \varsigma(\mathsf{at})\}_{\mathsf{at}\in\Omega_S}) = g^{\frac{r}{\prod_{\mathsf{at}\in\Omega_S}(\gamma+\varsigma(\mathsf{at}))}}$$

   using the algorithm $\mathsf{Aggregate}$ of [12]. From A_1, compute $T_1 = A_1^{\frac{1}{\prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}\varsigma(\mathsf{at})}}$.
2. Define the value $P_{(\Omega_S,S)}(\gamma)$ as

$$P_{(\Omega_S,S)}(\gamma) = \frac{1}{\gamma}\left(\prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S} (\gamma+\varsigma(\mathsf{at})) - \prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S} \varsigma(\mathsf{at}) \right).$$

   Since $|\Omega_S| = t$, the degree of the polynomial $P_{(\Omega_S,S)}(X)$ is $n-2$. Therefore, from SK_{Ω}, one can compute $h^{r\cdot P_{(\Omega_S,S)}(\gamma)/(\prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}\varsigma(\mathsf{at}))}$ and multiply this value with the last element in SK_{Ω}, which is $h^{\frac{r-1}{\gamma}}$, to obtain

$$T_2 = h^{\frac{r-1}{\gamma}} \cdot h^{r\frac{P_{(\Omega_S,S)}(\gamma)}{\prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})\setminus\Omega_S}\varsigma(\mathsf{at})}}.$$

   Note that the obtained values $T_1, T_2 \in \mathbb{G}$ satisfy the equality

$$e(T_2, u^{-1}) \cdot e\Big(T_1, h^{\alpha\cdot\prod_{\mathsf{at}\in(S\cup\mathcal{D}_{n+t-1-s})}(\gamma+\varsigma(\mathsf{at}))}\Big) \ = \ e(g^{\alpha}, h) \qquad (1)$$

   and that, in the terms in the left-hand-side of equality (1), the second argument of each pairing is publicly computable using pms and mpk.
3. Compute $M = m_1 \ldots m_k = H(\mathsf{Msg}, \Gamma) \in \{0,1\}^k$ and use M to form a message-specific Groth-Sahai CRS $\mathbf{g}_M = (\vec{g_1}, \vec{g_2}, \vec{g}_{3,M})$. Namely, for $i = 0$ to k, parse $\vec{g_{3,i}}$ as $(g_{X,i}, g_{Y,i}, g_{Z,i})^{\top} \in \mathbb{G}^3$. Then, define the vector $\vec{g}_{3,M} = \big(g_{X,0} \cdot \prod_{i=1}^{k} g_{X,i}^{m_i},\ g_{Y,0} \cdot \prod_{i=1}^{k} g_{Y,i}^{m_i},\ g_{Z,0} \cdot \prod_{i=1}^{k} g_{Z,i}^{m_i}\big)^{\top}$.

4. Using the newly defined $\mathbf{g}_M = (\vec{g_1}, \vec{g_2}, \vec{g}_{3,M})$, generate Groth-Sahai commitments to T_1 and T_2. Namely, pick $r_1, s_1, t_1, r_2, s_2, t_2 \xleftarrow{R} \mathbb{Z}_p$ and compute $\vec{C}_{T_j} = (1, 1, T_j)^\top \cdot \vec{g_1}^{r_j} \cdot \vec{g_2}^{s_j} \cdot \vec{g}_{3,M}^{\,t_j}$ for $j \in \{1, 2\}$. Then, generate a NIZK proof that committed variables (T_1, T_2) satisfy the pairing-product equation (1). To this end, we introduce an auxiliary variable $\Theta \in \mathbb{G}$ (with its own commitment $\vec{C}_\Theta = (1, 1, \Theta)^\top \cdot \vec{g_1}^{r_\theta} \cdot \vec{g_2}^{s_\theta} \cdot \vec{g}_{3,M}^{\,t_\theta}$, for $r_\theta, s_\theta, t_\theta \xleftarrow{R} \mathbb{Z}_p$), which takes on the value $\Theta = h$, and actually prove that

$$e(T_1, H_S) = e(g^\alpha, \Theta) \cdot e(T_2, u) \tag{2}$$
$$e(g, \Theta) = e(g, h), \tag{3}$$

where $H_S = h^{\alpha \cdot \prod_{\mathsf{at} \in (S \cup \mathcal{D}_{n+t-1-s})} (\gamma + \varsigma(\mathsf{at}))}$. The proofs for relations (2) and (3) are called $\vec{\pi_1}$ and $\vec{\pi_2}$, respectively, and they are given by

$$\vec{\pi}_1 = (\pi_{1,1}, \pi_{1,2}, \pi_{1,3})^\top = \left(H_S^{r_1} \cdot (g^\alpha)^{-r_\theta} \cdot u^{-r_2},\ H_S^{s_1} \cdot (g^\alpha)^{-s_\theta} \cdot u^{-s_2},\ H_S^{t_1} \cdot (g^\alpha)^{-t_\theta} \cdot u^{-t_2}\right)^\top$$
$$\vec{\pi}_2 = (\pi_{2,1}, \pi_{2,2}, \pi_{2,3})^\top = \left(g^{r_\theta},\ g^{s_\theta},\ g^{t_\theta}\right)^\top.$$

Finally, output the signature $\sigma = \left(\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{C}_\theta, \vec{\pi}_1, \vec{\pi_2}\right) \in \mathbb{G}^{15}$.

▶ $\mathsf{Verify}(\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}, \sigma, \Gamma)$: it first parses Γ as a pair (t, S) and σ as $\left(\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{C}_\theta, \vec{\pi}_1, \vec{\pi_2}\right)$. It computes $M = m_1 \ldots m_k = H(\mathsf{Msg}, \Gamma) \in \{0, 1\}^k$ and forms the corresponding vector

$$\vec{g}_{3,M} = \left(g_{X,0} \cdot \prod_{i=1}^{k} g_{X,i}^{m_i},\ g_{Y,0} \cdot \prod_{i=1}^{k} g_{Y,i}^{m_i},\ g_{Z,0} \cdot \prod_{i=1}^{k} g_{Z,i}^{m_i}\right)^\top \in \mathbb{G}^3.$$

Then, parse the proofs $\vec{\pi_1}$ and $\vec{\pi_2}$ as vectors $(\pi_{1,1}, \pi_{1,2}, \pi_{1,3})^\top$ and $(\pi_{2,1}, \pi_{2,2}, \pi_{2,3})^\top$, respectively. Define $H_S = h^{\alpha \cdot \prod_{\mathsf{at} \in (S \cup \mathcal{D}_{n+t-1-s})} (\gamma + \varsigma(\mathsf{at}))}$ and return 1 if the relations

$$E(H_S, \vec{C_{T_1}}) = E(g^\alpha, \vec{C_\theta}) \cdot E(u, \vec{C_{T_2}}) \cdot E(\pi_{1,1}, \vec{g_1}) \cdot E(\pi_{1,2}, \vec{g_2}) \cdot E(\pi_{1,3}, \vec{g}_{3,M}) \tag{4}$$
$$E(g, \vec{C_\theta}) = E\big(g, (1, 1, h)\big) \cdot E(\pi_{2,1}, \vec{g_1}) \cdot E(\pi_{2,2}, \vec{g_2}) \cdot E(\pi_{2,3}, \vec{g}_{3,M}) \tag{5}$$

are both satisfied. Otherwise, return 0.

Correctness. The correctness of the scheme immediately follows from the correctness of Groth-Sahai proofs.

An alternative scheme in the ROM. Steps 3 and 4 in the previous signing protocol can be replaced with a Σ-protocol, using the well-known Fiat-Shamir technique [15]. The result is a non-interactive zero-knowledge signature of knowledge for the message Msg, related to the knowledge of secret values T_1, T_2 satisfying equality (1). The resulting signatures would consist of 4 group elements, but the security of the scheme would rely on the assumption that the underlying hash function behaves as a random oracle.

3.1 Security Analysis

This first scheme is selective-predicate and adaptive-message unforgeable under chosen-message attacks, assuming the hardness of both the DLIN problem and the $(n - s^\star, n + t^\star - 1, t^\star + 1)$-aMSE-CDH problem, where $s^\star$ and $t^\star$ are the size of the attribute set and the threshold of the challenge signing policy. Computational privacy can be proven based on the hardness of the DLIN problem.

Theorem 1. *The scheme is selective-predicate and adaptive-message unforgeable under chosen-message attacks assuming that (1) H is a collision-resistant hash function; (2) the DLIN assumption holds in $\mathbb{G}$; (3) the $(n - s^\star, n + t^\star - 1, t^\star + 1)$-aMSE-CDH assumption holds in $(\mathbb{G}, \mathbb{G}_T)$, where n is the maximal number of attributes in the attribute set S and where $s^\star$ and $t^\star$ are the size of the attribute set and the threshold of the challenge signing policy.*

The proof of this theorem can be found in Appendix A.1.

Theorem 2. *This first ABS scheme enjoys computational privacy, assuming that the DLIN assumption holds in $\mathbb{G}$.*

Proof. (Sketch.) The proof consists in considering two games: Game_0 and Game_1. The first game, Game_0, is the real privacy game as described in Definition 5. In particular, when executing the trusted setup algorithm $\mathsf{ABS.TSetup}$, the challenger chooses the vectors $(\vec{g_1}, \vec{g_2}, \{\vec{g_{3,i}}\}_{i=0}^k)$ such that $\vec{g_{3,i}}$ is linearly dependent with $(\vec{g_1}, \vec{g_2})$, for all $i = 0, \ldots, k$. The only difference between Game_1 and Game_0 is that, in Game_1, the vector $\vec{g_{3,i}}$ is chosen at random so that it is linearly independent with $(\vec{g_1}, \vec{g_2})$, for all $i = 0, \ldots, k$. Groth-Sahai [17] proved that this change is indistinguishable, under the DLIN assumption. Finally, in Game_1, the only values that could leak any information about the subset of attributes held by the signer are $\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{\pi}_1$. But in the setting of Game_1, these commitments and proofs are perfectly hiding: they do not reveal any information about the committed values T_1, T_2. Therefore, privacy of the attributes holds unconditionally in Game_1. □

4 A Second Short Attribute-Based Signature Scheme for Threshold Predicates

The main advantage of our second ABS scheme over the previous one is that signatures are much shorter, since they have only three group elements. This comes at the cost of longer secret keys sk_Ω, containing $(2n+2) \times (|\Omega| + n)$ group elements. Another advantage is that the size of the considered universe of attributes may be much larger, even exponential in the security parameter λ; we only need that all attributes in the universe $\mathcal{P}$ are encoded as different elements of $\mathbb{Z}_p^*$.

▶ $\mathsf{TSetup}(\lambda, \mathcal{P}, n)$: chooses a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^k$, for some integer $k \in \mathsf{poly}(\lambda)$, as well as bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p > 2^\lambda$ with $g \xleftarrow{R} \mathbb{G}$. It also picks $u_0, u_1, \ldots, u_k \xleftarrow{R} \mathbb{G}$ and sets $\vec{U} = (u_0, u_1, \ldots, u_k)^\top$. It finally chooses a set $\mathcal{D} = \{d_1, \ldots, d_n\}$ of n distinct elements of $\mathbb{Z}_p$ that will serve as dummy attributes.
The resulting public parameters are $\mathsf{pms} = \Big(\mathcal{P},\ n,\ \lambda,\ \mathbb{G},\ \mathbb{G}_T, g,\ \vec{U},\ \mathcal{D},\ H\Big)$.

▶ $\mathsf{MSetup}(\mathsf{pms})$: randomly chooses $\alpha, \alpha_0 \xleftarrow{R} \mathbb{Z}_p$, $\vec{\alpha} = (\alpha_1, \ldots, \alpha_N)^\top \xleftarrow{R} \mathbb{Z}_p^N$, where $N = 2n + 1$. It then computes $e(g,g)^\alpha$, $h_0 = g^{\alpha_0}$, $\vec{H} = (h_1, \ldots, h_N)^\top = g^{\vec{\alpha}}$.
The master secret key is defined to be $\mathsf{msk} = g^\alpha$ and the master public key is $\mathsf{mpk} = \Big(e(g,g)^\alpha,\ h_0,\ \vec{H}\Big)$.

▶ $\mathsf{Keygen}(\mathsf{pms}, \mathsf{mpk}, \mathsf{msk}, \Omega)$: to generate a key for the attribute set Ω, the algorithm first chooses an implicit polynomial $Q_\Omega[X] = \alpha + \beta_1 X + \cdots + \beta_{n-1} X^{n-1}$ for random coefficients $\beta_1, \ldots, \beta_{n-1} \xleftarrow{R} \mathbb{Z}_p$. Then, it proceeds as follows.

1. For each attribute $\omega \in \Omega$, choose a random exponent $r_\omega \xleftarrow{R} \mathbb{Z}_p$ and generate a key component $\mathsf{SK}_\omega = (D_{\omega,1}, D_{\omega,2}, K_{\omega,1}, \ldots, K_{\omega,N-1})$ where

$$D_{\omega,1} = g^{Q_\Omega(\omega)} \cdot h_0^{r_\omega}, \qquad D_{\omega,2} = g^{r_\omega}, \qquad \Big\{K_{\omega,i} = \big(h_1^{-\omega^i} \cdot h_{i+1}\big)^{r_\omega}\Big\}_{i=1,\ldots,N-1}. \tag{6}$$

2. For each $d \in \mathcal{D}$, generate a private key component $\mathsf{SK}_d = (D_{d,1}, D_{d,2}, K_{d,1}, \ldots, K_{d,N-1})$ in the same way as in (6), by choosing a fresh random value $r_d \in \mathbb{Z}_p$ and computing

$$D_{d,1} = g^{Q_\Omega(d)} \cdot h_0^{r_d}, \qquad D_{d,2} = g^{r_d}, \qquad \Big\{K_{d,i} = \big(h_1^{-w^i} \cdot h_{i+1}\big)^{r_d}\Big\}_{i=1,\ldots,N-1}. \tag{7}$$

The private key finally consists of $SK_\Omega = \big(\{\mathsf{SK}_\omega\}_{\omega\in\Omega}, \{\mathsf{SK}_d\}_{d\in\mathcal{D}}\big)$.

▶ $\mathsf{Sign}(\mathsf{pms}, \mathsf{mpk}, SK_\Omega, \mathsf{Msg}, \Gamma)$: to sign $\mathsf{Msg} \in \{0,1\}^*$ w.r.t. the policy $\Gamma = (t, S)$, where S is an attribute set of size $s = |S| \leq n$ and $t \in \{1, \ldots, s\}$, the algorithm first computes $M = H(\mathsf{Msg}, \Gamma) \in \{0,1\}^k$ and parses the private key SK_Ω as $\big(\{\mathsf{SK}_\omega\}_{\omega\in\Omega}, \{\mathsf{SK}_d\}_{d\in\mathcal{D}}\big)$.

1. It considers the subset $\mathcal{D}_{n-t} \subset \mathcal{D}$ containing the $n-t$ first attributes of $\mathcal{D}$ (chosen in some pre-specified lexicographical order). It also chooses an arbitrary subset $S_t \subset \Omega \cap S$ such that $|S_t| = t$ and defines $\vec{Y} = (y_1, \ldots, y_N)^\top$ as the vector containing the coefficients of the polynomial

$$P_S(Z) = \sum_{i=1}^{n-t+s+1} y_i Z^{i-1} = \prod_{\omega\in S}(Z-\omega) \cdot \prod_{d\in\mathcal{D}_{n-t}} (Z-d). \tag{8}$$

   Since $n - t + s + 1 \leq 2n + 1 = N$, the coordinates $y_{n-t+s+2}, \ldots, y_N$ are all set to 0.
2. For each $\omega \in S_t$, use $\mathsf{SK}_\omega = (D_{\omega,1}, D_{\omega,2}, \{K_{\omega,i}\}_{i=1}^{N-1})$ to compute

$$D'_{\omega,1} = D_{\omega,1} \cdot \prod_{i=1}^{N-1} K_{\omega,i}^{y_{i+1}} = g^{Q_\Omega(\omega)} \cdot \big(h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}\big)^{r_\omega}. \tag{9}$$

   The last equality comes from the fact that $P_S(\omega) = 0$ for all $\omega \in S$.
3. Likewise, for each dummy attribute $d \in \mathcal{D}_{n-t}$, use $\mathsf{SK}_d = (D_{d,1}, D_{d,2}, \{K_{d,i}\}_{i=1}^{N-1})$ to compute

$$D'_{d,1} = D_{d,1} \cdot \prod_{i=1}^{N-1} K_{d,i}^{y_{i+1}} = g^{Q_\Omega(d)} \cdot \big(h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}\big)^{r_d}. \tag{10}$$

4. Using the values $\{D'_{\omega,1}\}_{\omega\in S_t}$ and $\{D'_{d,1}\}_{d\in\mathcal{D}_{n-t}}$ and the corresponding $D_{\omega,2} = g^{r_w}$, $D_{d,2} = g^{r_d}$, compute

$$D_1 = \prod_{\omega\in S_t} {D'_{\omega,1}}^{\Delta_\omega^{S_t\cup\mathcal{D}_{n-t}}(0)} \cdot \prod_{d\in\mathcal{D}_{n-t}} {D'_{d,1}}^{\Delta_d^{S_t\cup\mathcal{D}_{n-t}}(0)} = g^\alpha \cdot (h_0 \cdot \prod_{i=1}^{N} h_i^{y_i})^r \tag{11}$$

$$D_2 = \prod_{\omega\in S_t} {D_{\omega,2}}^{\Delta_\omega^{S_t\cup\mathcal{D}_{n-t}}(0)} \cdot \prod_{d\in\mathcal{D}_{n-t}} {D_{d,2}}^{\Delta_d^{S_t\cup\mathcal{D}_{n-t}}(0)} = g^r, \tag{12}$$

   where $r = \sum_{\omega\in S_t} \Delta_\omega^{S_t\cup\mathcal{D}_{n-t}}(0) \cdot r_\omega + \sum_{d\in\mathcal{D}_{n-t}} \Delta_d^{S_t\cup\mathcal{D}_{n-t}}(0) \cdot r_d$.
5. Parse $M \in \{0,1\}^k$ as a string $m_1 \ldots m_k$ where $m_j \in \{0,1\}$ for $j = 1, \ldots, k$. Then, choose $z, w \xleftarrow{R} \mathbb{Z}_p$ and compute

$$\sigma_1 = D_1 \cdot \big(h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}\big)^w \cdot \big(u_0 \cdot \prod_{j=1}^{k} u_j^{m_j}\big)^z, \qquad \sigma_2 = D_2 \cdot g^w, \qquad \sigma_3 = g^z.$$

Return the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3) \in \mathbb{G}^3$.

▶ Verify($\mathsf{pms}, \mathsf{mpk}, \mathsf{Msg}, \sigma, \Gamma$): it first parses Γ as a pair (t, S). It computes $M = H(\mathsf{Msg}, \Gamma) \in \{0,1\}^k$ and considers the subset $\mathcal{D}_{n-t} \subset \mathcal{D}$ containing the $n-t$ first dummy attributes of $\mathcal{D}$. Then, it defines the vector $\vec{Y} = (y_1, \ldots, y_N)^\top$ from the polynomial $P_S(Z)$ as per (8). The protocol accepts the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ as valid and thus outputs 1 if

$$e(g,g)^\alpha = \frac{e(\sigma_1, g)}{e(\sigma_2, h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}) \cdot e(\sigma_3, u_0 \cdot \prod_{j=1}^{k} u_j^{m_j})}. \tag{13}$$

Otherwise, it outputs 0.

Correctness. The correctness of the scheme follows from the property that for each attribute $\omega \in S_t \subset S \cap \Omega$, the vector $\vec{X}_\omega^N = (1, \omega, \omega^2, \ldots, \omega^{N-1})$ is orthogonal to $\vec{Y}$, so that we have

$$D'_{\omega,1} = D_{\omega,1} \cdot \prod_{i=1}^{N-1} K_{\omega,i}^{y_{i+1}} = g^{Q_\Omega(\omega)} \cdot \Big(h_0 \cdot h_1^{-(\langle \vec{X}_\omega^N, \vec{Y} \rangle - y_1)} \prod_{i=2}^{N} h_i^{y_i}\Big)^{r_\omega} = g^{Q_\Omega(\omega)} \cdot \Big(h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}\Big)^{r_\omega},$$

which explains the second equality of (9) and the same holds for (10). In addition, the values (D_1, D_2) obtained as per (11)-(12) satisfy $e(D_1, g) = e(g,g)^\alpha \cdot e(h_0 \cdot \prod_{i=1}^{N} h_i^{y_i}, D_2)$, which easily leads to the verification equation (13).

4.1 Security Analysis

Our second scheme is selective-predicate and adaptive-message unforgeable under chosen-message attacks, by reduction to the hardness of the n-Diffie-Hellman Exponent (n-DHE) problem ([8]). This scheme also enjoys unconditional privacy, which is another advantage over our first scheme.

Theorem 3. *The scheme is selective-predicate and adaptive-message unforgeable under chosen-message attacks if H is a collision-resistant hash function and if the $(2n+1)$-DHE assumption holds in $\mathbb{G}$, where n is the maximal number of attributes in the attribute set S.*

The proof of this theorem can be found in Appendix A.2.

Theorem 4. *This second ABS scheme enjoys perfect privacy.*

Proof. A valid signature for the threshold policy (t, S) which was produced using the subset of attributes $S_t \subset S$, $|S_t| = t$ and with randomness w can also be produced for any other set $S'_t \subset S$, $|S'_t| = t$ with randomness w'. More specifically, if $r = \sum_{\omega \in S_t} \Delta_\omega^{S_t \cup \mathcal{D}_{n-t}}(0) \cdot r_\omega + \sum_{d \in \mathcal{D}_{n-t}} \Delta_d^{S_t \cup \mathcal{D}_{n-t}}(0) \cdot r_d$ and $r' = \sum_{\omega \in S'_t} \Delta_\omega^{S_t \cup \mathcal{D}_{n-t}}(0) \cdot r_\omega + \sum_{d \in \mathcal{D}_{n-t}} \Delta_d^{S_t \cup \mathcal{D}_{n-t}}(0) \cdot r_d$, any pair (w, w') satisfying $r + w = r' + w'$ will result in the same signature for S_t and S'_t. □

5 More General Signing Predicates

Our schemes admit some extensions to deal with more general monotone predicates. In general, a predicate is a pair (S, Γ), where $S = \{\mathsf{at}_1, \ldots, \mathsf{at}_s\}$ is a set of attributes and $\Gamma \subset 2^S$ is a monotone increasing family of subsets of S. An attribute-based signature for a pair (S, Γ) convinces the verifier that the signer holds some subset of attributes $A \in \Gamma$, without revealing any information on A.

5.1 Extensions for the First Scheme

Similarly to what is suggested in [18], our first signature scheme can be extended to admit weighted threshold predicates. A pair (S, Γ) is a weighted threshold predicate if there exist a threshold t and an assignment of weights $\omega : S \to \mathbb{Z}^+$ such that $\Omega \in \Gamma \iff \sum_{\mathsf{at}\in\Omega} \omega(\mathsf{at}) \geq t$. If K is an upper bound for $\omega(\mathsf{at})$, for all attributes $\mathsf{at} \in \mathcal{P}$ and all possible assignments of weights that realize weighted threshold predicates, the idea is to consider an augmented universe of attributes $\mathcal{P}' = \{\mathsf{at}_1||1, \mathsf{at}_1||2, \ldots, \mathsf{at}_1||K, \ldots, \mathsf{at}_m||1, \ldots, \mathsf{at}_m||K\}$. The size of each user's secret key is just increased by a factor of K. To sign a message for the weighted threshold predicate (S, Γ), where Γ is defined by ω and t, the signer can use the threshold signature routine of our first scheme with the threshold t and attribute set $S' = \{\mathsf{at}_1||1, \ldots, \mathsf{at}_1||\omega(\mathsf{at}_1), \ldots, \mathsf{at}_s||1, \ldots, \mathsf{at}_s||\omega(\mathsf{at}_s)\}$. If the user holds a subset of attributes $\Omega \in \Gamma$, he will have $\omega(\mathsf{at})$ valid elements in his secret key, for each attribute $\mathsf{at} \in \Omega$; and since $\sum_{\mathsf{at}\in\Omega} \omega(\mathsf{at}) \geq t$, he will be able to run the threshold signing protocol.

Furthermore, since the final form of the signatures in our first threshold scheme is that of a Groth-Sahai non-interactive proof, one could consider signing predicates which are described by a monotone formula (OR / AND gates) over threshold clauses. The Groth-Sahai proof would be then a proof of knowledge of some values that satisfy a monotone formula of equations. The size of such a proof (and therefore, the size of the resulting attribute-based signatures) would be linear in the number of threshold clauses in the formula. We stress that this is still better than having size linear in the number of involved attributes, as in all previous constructions.

5.2 Extensions for the Second Scheme

The idea of our second scheme is that a (threshold) attribute-based signature can be computed only if the signer holds t attributes in S which, combined with $n - t$ dummy attributes, lead to n attributes at such that $P_S(\mathsf{at}) = 0$. This makes it possible to interpolate a polynomial $Q_\Omega(X)$ with degree $n - 1$, recover in some way the value g^α and produce a valid signature. To admit any possible value of the threshold t in $\{1, \ldots, n\}$, the number of dummy attributes must be n. The key generation protocol for a subset of attributes Ω already restricts the usability of the signature protocol to threshold predicates. But this can be extended to admit some more general families of predicates, specifically for those that can be realized with a secret sharing scheme with similar properties to Shamir's threshold secret sharing scheme. We give here two examples of such families of predicates. The ideas underlying this extension are somehow related to those in [11], where dummy attributes were used to design attribute-based encryption schemes for general decryption predicates.

Hierarchical Threshold Predicates. We follow the definitions and notations of hierarchical threshold families that appear in [31]. The set of involved attributes is divided in c disjoint levels, $S = S_1 \cup S_2 \cup \ldots \cup S_c$, where $c \geq 1$ is an integer. A predicate $\Gamma \subset 2^S$ is hierarchical threshold if there exists a strictly increasing sequence of integers $0 < k_1 < k_2 < \ldots < k_c$ such that

$$\Gamma = \{\Omega \subset S \ : \ \left|\Omega \cap \left(\bigcup_{i=1}^{\ell} S_i\right)\right| \geq k_\ell, \ \forall \ell \in \{1, 2, \ldots, c\}\}$$

That is, a subset is authorized if it contains at least k_1 attributes from the first level, and at least k_2 players from the two first levels, and so on. Tassa constructed in [31] a secret sharing scheme realizing this kind of predicates, using Birkhoff interpolation: the dealer chooses a polynomial $f(X)$

with degree $k_c - 1$, the secret is $\alpha = f(0)$ and the share of an attribute $\mathsf{at}_{i,j} \in S_i$ from level i is $\alpha_{i,j} = f^{(k_{i-1})}(j)$, the k_{i-1} derivative of f evaluated in j.

To extend our second attribute-based signature scheme so that the admitted signing predicates are exactly those hierarchical threshold predicates with c levels, let n be the size of the universe of attributes $\mathcal{P} = \{\mathsf{at}_1, \ldots, \mathsf{at}_n\}$. In the Setup phase, n dummy attributes must be placed in each level $i = 1, \ldots, c$. Therefore, there will be cn dummy attributes in total. In the Key Generation phase, for a subset of attributes Ω, the master entity will use Tassa's secret sharing scheme for a hierarchical threshold predicate Γ' defined by the sequence $0 < n < 2n < \ldots < cn$. That is, the master entity chooses a fresh polynomial $Q_\Omega(X)$ of degree $cn-1$ such that $Q_\Omega(0) = \alpha$. The master entity will compute the shares for the attributes in Ω and for the cn dummy attributes (placed in different levels) to derive the values $D_{\omega,1}, D_{d,1}$ that appear in SK_Ω. Finally, if a user wants to sign for a hierarchical threshold predicate (S, Γ) defined by the sequence $0 < k_1 < k_2 < \ldots < k_c$, where $k_c \leq n$, the polynomial $P_S(Z)$ is defined from the set S and from $n - k_i$ dummy attributes in level i, for $i = 1, \ldots, c$. The user can combine his attributes (in S) with $n - k_i$ dummy attributes in level i, for $i = 1, \ldots, c$, to interpolate from SK_Ω the polynomial $Q_\Omega(X)$, in the exponent, and recover the value product containing factor g^α that is needed to sign. The form of the signatures, the Verification phase and the security analysis are almost identical as in our threshold scheme in Section 4.

The extension is limited in the sense that the number of levels c in the admitted hierarchical threshold policies is set at the beginning. If more flexibility is desired, both the Setup and the Key Generation processes should be repeated in parallel for any desired number of levels, at the cost of an increase in the length of the public parameters and the secret keys.

Compartmented Access Structures. Compartmented families were introduced in [10]. Again, the set of involved attributes is partitioned: $S = S_1 \cup S_2 \cup \ldots \cup S_c$, with $S_{i_1} \cap S_{i_2} = \emptyset$ for all $i_1 \neq i_2$. A predicate $\Gamma \subset 2^S$ is compartmented if there exists a sequence of thresholds $t; t_1, t_2, \ldots, t_c$ such that

$$\Gamma = \{\Omega \subset S \ : \ |\Omega| \geq t \text{ and } |\Omega \cap S_\ell| \geq t_\ell, \ \forall \ell \in \{1, 2, \ldots, c\}\}$$

Such a predicate makes sense only when $t \leq |S|$ and $t_\ell \leq |S_\ell|$, for $\ell = 1, \ldots, c$. We can assume $t \geq t_1 + \ldots + t_c$; otherwise, replacing t with $t' = t_1 + \ldots + t_c$ and obtain the same predicate.

Brickell constructed in [10] a vector space secret sharing scheme to realize a compartmented predicate (S, Γ) defined by a sequence of thresholds $t; t_1, t_2, \ldots, t_c$. The dealer is associated to the vector $\vec{v}_0 = (1, 0, \ldots, 0)$ and then each attribute $\mathsf{at}_{i,j} \in S_j$ in compartment j is associated to a vector $\vec{v}_{i,j}$ in such a way that $\Omega \in \Gamma$ if and only if $\vec{v}_0 \in \langle \{\vec{v}_{i,j}\}_{\mathsf{at}_{i,j} \in \Omega} \rangle$ (in other words, if the vectors associated to the attributes in Ω span the vector $\vec{v}_0$). To distribute a secret α, the dealer chooses a random vector $\vec{a}$ such that $\vec{a} \cdot \vec{v}_0 = \alpha$, and assigns to each attribute $\mathsf{at}_{i,j}$ the share $\alpha_{i,j} = \vec{a} \cdot \vec{v}_{i,j}$. The secret α can be easily obtained from the shares of any subset of attributes $\Omega \in \Gamma$.

Similarly, our second attribute-based signature scheme can be extended to admit compartmented predicates with a fixed number, c, of compartments. If n is the size of the universe of attributes, n dummy attributes must be placed in each of the compartments during the Setup. This means we need cn dummy attributes in total, again. In the Key Generation for a subset of attributes Ω, the master entity will use Brickell's secret sharing scheme for the compartmented predicate Γ' defined by sequence of thresholds $cn; n, n, \ldots, n$. The shares corresponding to the attributes in Ω and to the cn dummy attributes, according to the compartment where they belong, will be computed and used to get the values $D_{\omega,1}, D_{d,1}$ in SK_Ω. When a user wants to sign for such a predicate (S, Γ) defined by the sequence of thresholds $t; t_1, t_2, \ldots, t_c$, where $t \leq n$, he combines his real attributes (satisfying Γ) with $n - t_i$ dummy attributes in the compartment i, for $i = 1, \ldots, c$.

The resulting subset of attributes will be authorized for the predicate Γ', and so the distributed secret α (or a product containing the factor g^α, as it is needed in our signature scheme) can be recovered from the shares of the user. The polynomial $P_S(Z)$ (see equation (8) in Section 4) is again defined from the set S and from $n - t_i$ dummy attributes in compartment i, for $i = 1, \ldots, c$.

6 Concluding Remarks

We have proposed the first two threshold ABS schemes which produce constant-size signatures. The signatures consist respectively of 15 and 3 group elements, whereas the secret keys sk_Ω contain respectively $|\Omega| + n$ and $(2n + 2) \times (|\Omega| + n)$ group elements. The security of both schemes is proven in the standard model with respect to assumptions related to bilinear groups. Note that the considered unforgeability model is not the strongest possible, since the attacker has to choose the signing policy he wants to attack at the beginning of the unforgeability game.

References

1. N. Attrapadung, B. Libert, E. de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *PKC'11*, *LNCS* 6571, pp. 90–108, 2011.
2. J. Bethencourt, A. Sahai, B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy (S&P'07)*, IEEE Society Press, pp. 321-334, 2007.
3. D. Boneh, X. Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In *Eurocrypt'04*, *LNCS* 3027, pp. 223–238, 2004.
4. D. Boneh, X. Boyen, E.-J. Goh. Hierarchical identity-based encryption with constant size ciphertext. In *Eurocrypt'05*, *LNCS* 3494, pp. 440–456, 2005.
5. D. Boneh, X. Boyen and H. Shacham. Short group signatures. In *Crypto'04*, *LNCS* 3152, pp. 41–55, 2004.
6. X. Boyen. Mesh signatures. In *Eurocrypt'07*, *LNCS* 4515, pp. 210–227, 2007.
7. E. Bresson, J. Stern and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Crypto'02*, *LNCS* 2442, pp. 465–480, 2002.
8. D. Boneh, C. Gentry, B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Crypto'05*, *LNCS* 3621, pp. 258–275, 2005.
9. D. Boneh, M. Hamburg. Generalized identity based and broadcast encryption schemes. In *Asiacrypt'08*, *LNCS* 5350, pp. 455–470, 2008.
10. E.F. Brickell. Some ideal secret sharing schemes. In *Journal of Combinatorial Mathematics and Combinatorial Computing*, Volume 9, pp. 105–113, 1989.
11. V. Daza, J. Herranz, P. Morillo, C. Ràfols. Extended access structures and their cryptographic applications. In *Applicable Algebra in Engineering, Communication and Computing*, Volume 21, Issue 4, pp. 257–284, 2010.
12. C. Delerablée, D. Pointcheval. Dynamic threshold public-key encryption. In *Crypto'08*, *LNCS* 5157, pp. 317–334, 2008.
13. K. Emura, A. Miyaji, A. Nomura, K. Omote, M. Soshi. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In *ISPEC '09*, *LNCS* 5451, pp. 13–23, 2009.
14. A. Escala, J. Herranz, P. Morillo. Revocable attribute-based signatures with adaptive security in the standard model. In *Africacrypy'11*, *LNCS* 6737, pp. 224–241, 2011.
15. A. Fiat, A. Shamir. How to prove yourself: practical solutions of identification and signature problems. In *Crypto'86*, *LNCS* 263, pp. 186–194, 1986.
16. V. Goyal, O. Pandey, A. Sahai, B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS'06*, ACM Press, pp. 89–98, 2006.
17. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Eurocrypt'08*, LNCS 4965, pp. 415–432, 2008.
18. J. Herranz, F. Laguillaumie, C. Ràfols. Constant-size ciphertexts in threshold attribute-based encryption. In *PKC'10*, *LNCS* 6056, pp. 19–34, 2010.
19. D. Hofheinz, E. Kiltz. Programmable hash functions and their applications. In *Crypto'08*, *LNCS* 5157, pp. 21–38, 2008.
20. A. Lewko, T. Okamoto, A. Sahai, K. Takashima, B. Waters. Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In *Eurocrypt'10*, *LNCS* 6110, pp. 62–91, 2010.

21. J. Li, M.H. Au, W. Susilo, D. Xie, K. Ren. Attribute-based signature and its applications. In *ASIACCS'10*, ACM Press, pp. 60–69, 2010.
22. J. Li and K. Kim. Hidden attribute-based signatures without anonymity revocation. In *Information Sciences*, 180 (9), pp. 1681–1689, 2010.
23. T. Malkin, I. Teranishi, Y. Vahlis, M. Yung. Signatures resilient to continual leakage on memory and computation. In *TCC'11*, *LNCS* 6597, pp. 89–106, 2011.
24. H.K. Maji, M. Prabhakaran, M. Rosulek. Attribute-based signatures. In *CT-RSA'11*, *LNCS* 6558, pp. 376–392, 2011.
25. M. Naor. On cryptographic assumptions and challenges. In *Crypto'03*, *LNCS* 2729, pp. 96–109, 2003.
26. T. Okamoto and K. Takashima. Homomorphic encryption and signatures from vector decomposition. In *Pairing'08*, *LNCS* 5209, pp. 57–74, 2008.
27. T. Okamoto, K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *PKC'11*, *LNCS* 6571, pp. 35–52, 2011.
28. R. L. Rivest, A. Shamir and Y. Tauman. How to leak a secret. In *Asiacrypt'01*, *LNCS* 2248, pp. 552-565, 2001.
29. A. Sahai, B. Waters. Fuzzy identity-based encryption In *Eurocrypt'05*, *LNCS* 3494, pp. 457–473, 2005.
30. S.F. Shahandashti, R. Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. In *Africacrypt'09*, *LNCS* 5580, pp. 198–216, 2009.
31. T. Tassa. Hierarchical threshold secret sharing. In *Journal of Cryptology*, 20 (2), pp. 237–264, 2007.
32. B. Waters. Efficient identity-based encryption without random oracles. In *Eurocrypt'05*, *LNCS* 3494, pp. 114–127, 2005.

A Unforgeability Proofs

In this appendix we detail the unforgeability proofs of the two ABS schemes proposed in this work.

A.1 Proof of Theorem 1

We prove the theorem by considering a modified distribution of vectors $(\vec{g_1}, \vec{g_2}, \{\vec{g_{3,i}}\}_{i=0}^{k})$ that are used to generate Groth-Sahai reference strings.

Namely, to prepare $\vec{g_1}$, $\vec{g_2}$ and $\{\vec{g_{3,i}}\}_{i=0}^{k}$, we randomly pick $\nu \xleftarrow{R} \{0,\ldots,k\}$, $\xi_{0,1}, \xi_{1,1}, \ldots, \xi_{k,1} \xleftarrow{R} \mathbb{Z}_p$, $\xi_{0,2}, \xi_{1,2}, \ldots, \xi_{k,2} \xleftarrow{R} \mathbb{Z}_p$ and $\rho_0, \rho_1, \ldots, \rho_k \xleftarrow{R} \{0,\ldots,\tau-1\}$, where $\tau = 2q$ (where q is the number of signing queries). We first set $\vec{g_1} = (g_1, 1, g)^\top$ and $\vec{g_2} = (1, g_2, g)^\top$, where $g_1 = g^a$ and $g_2 = g^b$ for randomly chosen $a, b \xleftarrow{R} \mathbb{Z}_p$. We then defines $\{\vec{g_{3,i}}\}_{i=0}^{k}$ by setting $\vec{g_{3,0}} = \vec{g_1}^{\xi_{0,1}} \cdot \vec{g_2}^{\xi_{0,2}} \cdot ((1,1,g)^\top)^{\nu\cdot\tau-\rho_0}$ as well as $\vec{g_{3,i}} = \vec{g_1}^{\xi_{i,1}} \cdot \vec{g_2}^{\xi_{i,2}} \cdot ((1,1,g)^\top)^{-\rho_i}$ for $i = 1,\ldots,k$.

The distribution of $\{\vec{g_{3,i}}\}_{i=0}^{k}$ is obviously different from its distribution in the real scheme. However, we argue that, if this modification in the distribution of public parameters noticeably affects the adversary's probability of success, the DLIN assumption can be broken. This statement if proved by lemma 1.

In a second step, lemma 2 proves that, with the above way of generating $\left(\vec{g_1}, \vec{g_2}, \{g_{3,i}\}_{i=0}^{k}\right)$, a successful forger $\mathcal{F}$ implies either a PPT algorithm to find collisions on H, a distinguisher for the DLIN problem or an algorithm $\mathcal{B}$ that solves the $(n - s^\star, n + t^\star - 1, t^\star + 1)$-aMSE-CDH problem.

Lemma 1. *Under the above distribution of $\left(\vec{g_1}, \vec{g_2}, \{\vec{g_{3,i}}\}_{i=0}^{k}\right)$, $\mathcal{F}$'s advantage is negligibly close to its advantage in the real scheme if the DLIN assumption holds in $\mathbb{G}$.*

Proof. The proof uses a sequence of $2k+2$ games Game_{real}, Game_0, Game'_0, Game_1, Game'_1, ..., Game_k, Game'_k which proceed as follows.

Game_{real}: is a game where the adversary is given vectors $(\vec{g_1}, \vec{g_2}, \{\vec{g}_{3,i}\}_{i=0}^{k})$ as in the real scheme.
Game_0: is a game where the vectors $\vec{g_1} = (g_1, 1, g)^\top$ and $\vec{g_2} = (1, g_2, g)^\top$ are chosen as in Game_{real} but $\vec{g}_{3,0}$ is chosen as $\vec{g}_{3,0} = \vec{g_1}^{\xi_{0,1}} \cdot \vec{g_2}^{\xi_{0,2}} \cdot ((1,1,g)^\top)^{\xi_{0,3}}$ for random $\xi_{0,1}, \xi_{0,2}, \xi_{0,3} \xleftarrow{R} \mathbb{Z}_p$. Vectors $\{\vec{g}_{3,i}\}_{i=1}^{k}$ are still chosen as in Game_{real}.

Game'_0: is identical to Game_0 with a modification in the distribution of the vector $\vec{g}_{3,0}$ which is now given by $\vec{g}_{3,0} = \vec{g_1}^{\xi_{0,1}} \cdot \vec{g_2}^{\xi_{0,2}} \cdot ((1,1,g)^\top)^{\nu \cdot \tau - \rho_0}$, for some $\nu \xleftarrow{R} \{0,\ldots,k\}$ and $\rho_0 \xleftarrow{R} \{0,\ldots,\tau-1\}$, while the distribution of $\{\vec{g}_{3,i}\}_{i=1}^k$ remains the same as in Game_{real}.

Game_j $(1 \leq j \leq k)$: is identical to Game'_{j-1} but the distribution of the vector $\vec{g}_{3,j}$ is changed into $\vec{g}_{3,j} = \vec{g_1}^{\xi_{j,1}} \cdot \vec{g_2}^{\xi_{j,2}} \cdot ((1,1,g)^\top)^{\xi_{j,3}}$ for random $\xi_{j,1}, \xi_{j,2}, \xi_{j,3} \xleftarrow{R} \mathbb{Z}_p$. As for vectors $\{\vec{g}_{3,i}\}_{i=j+1}^k$, they still live in the span of $\vec{g_1}$ and $\vec{g_2}$ as in Game_{real}.

Game'_j $(1 \leq j \leq k)$: is like Game_j but, instead of being a truly random vector of $\mathbb{G}^3$, $\vec{g}_{3,j}$ is now obtained as $\vec{g}_{3,j} = \vec{g_1}^{\xi_{j,1}} \cdot \vec{g_2}^{\xi_{j,2}} \cdot ((1,1,g)^\top)^{-\rho_i}$ for some $\rho_i \xleftarrow{R} \{0,\ldots,\tau-1\}$.

In Game'_k, the distribution of $\big(\vec{g_1}, \vec{g_2}, \{g_{3,i}\}_{i=0}^k\big)$ is clearly identical to the one suggested in the proof of theorem 1 and we just have to argue that all these games are computationally indistinguishable under the DLIN assumption. We first explain how the indistinguishability of Game_1, Game'_1, ..., Game_k, Game'_k can be proved.

Claim 1. Under DLIN, no PPT adversary can distinguish Game_j from Game'_{j-1} for all $1 \leq j \leq k$.

Proof. We construct a distinguisher $\mathcal{B}^{\mathsf{DLIN}}$ that takes as input $(g, g_1, g_2, g_1^{\delta_1}, g_2^{\delta_2}, \chi)$, for some $\delta_1, \delta_2 \xleftarrow{R} \mathbb{Z}_p$, with the goal of deciding if $\chi = g^{\delta_1+\delta_2}$ or $\chi \in_R \mathbb{G}$. To this end, $\mathcal{B}^{\mathsf{DLIN}}$ first defines $\vec{g_1} = (g_1, 1, g)^\top$, $\vec{g_2} = (1, g_2, g)^\top$. It generates $\vec{g}_{3,0}, \ldots, \vec{g}_{3,j-1}$ as Game'_{j-1} and $\vec{g}_{3,j+1}, \ldots, \vec{g}_{3,k}$ as in Game_{real}. As for $\vec{g}_{3,j}$, algorithm $\mathcal{B}^{\mathsf{DLIN}}$ defines it as $\vec{g}_{3,j} = (g_1^{\delta_1}, g_2^{\delta_2}, \chi)^\top$. It is clear that, if $\chi = g^{\delta_1+\delta_2}$, the adversary is playing Game'_{j-1} whereas, if $\chi \in_R \mathbb{G}$, it is playing Game_j. At the end of the attack game, the distinguisher $\mathcal{B}^{\mathsf{DLIN}}$ outputs 1 if the forger $\mathcal{F}$ manages to create a valid forgery and 0 otherwise.

Claim 2. Under DLIN, Game'_j is computationally indistinguishable from Game_j for all $1 \leq j \leq k$.

Proof. We show a distinguisher $\mathcal{B}^{\mathsf{DLIN}}$ that takes as input a tuple $(g, g_1, g_2, g_1^{\delta_1}, g_2^{\delta_2}, \chi)$ and decides if $\chi = g^{\delta_1+\delta_2}$ or $\chi \in_R \mathbb{G}$. To this end, $\mathcal{B}^{\mathsf{DLIN}}$ first defines $\vec{g_1} = (g_1, 1, g)^\top$, $\vec{g_2} = (1, g_2, g)^\top$. Vectors $\vec{g}_{3,0}$, $\vec{g}_{3,1}$, ..., $\vec{g}_{3,j-1}$ are then chosen as in Game'_{j-1} while vectors $\vec{g}_{3,j+1}$, ..., $\vec{g}_{3,k}$ are selected as in the real game Game_{real}. To prepare the vector $\vec{g}_{3,j}$, the distinguisher $\mathcal{B}^{\mathsf{DLIN}}$ randomly picks $\rho_i \xleftarrow{R} \{0,\ldots,\tau-1\}$ and computes $\vec{g}_{3,j} = (g_1^{\delta_1}, g_2^{\delta_2}, \chi \cdot g^{-\rho_i})^\top$. It is easy to see that, if $\chi \in_R \mathbb{G}$, algorithm $\mathcal{B}^{\mathsf{DLIN}}$ is playing Game_j with the forger $\mathcal{F}$ since $\vec{g}_{3,j}$ has the same distribution no matter if χ is multiplied by $g^{-\rho_i}$ or not. If $\chi = g^{\delta_1+\delta_2}$, $\mathcal{B}^{\mathsf{DLIN}}$ is rather playing Game'_j with $\mathcal{F}$. At the end of its interaction with $\mathcal{F}$, $\mathcal{B}^{\mathsf{DLIN}}$ outputs 1 if $\mathcal{F}$ outputs a successful forgery and 0 otherwise.

The computational indistinguishability of Game_0, Game'_0 and Game_1 is proved in the same way. The only change is that, during the transition from Game_0 to Game'_0, χ is multiplied by $g^{\nu \cdot \tau - \rho_0}$. Finally, the indistinguishability of Game_{real} and Game_0 is established exactly in the same way as that of Game'_{j-1} and Game_j, for $j \in \{1,\ldots,k\}$, in the proof of Claim 1. □

Lemma 2. *Under the modified distribution of $\big(\vec{g_1}, \vec{g_2}, \{g_{3,i}\}_{i=0}^k\big)$, $\mathcal{F}$'s advantage is negligible if the $(n - s^\star, n + t^\star - 1, t^\star + 1)$-aMSE-CDH assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ and if H is a collision-resistant hash function.*

Proof. We denote by $\mathcal{I}(\overrightarrow{x}_{2n+t^*-s^*-1}, \kappa, \alpha, \gamma, \omega, T)$ the input of the algorithm $\mathcal{B}$ that we design to solve the aMSE-CDH problem (recall Definition 1 in Section 2.2). At the outset of the game, the adversary $\mathcal{F}$ declares its chosen challenge policy $\Gamma^\star = (t^\star, S^\star)$, with $s^\star = |S^\star|$. Using this information, $\mathcal{B}$ chooses $\big(\vec{g_1}, \vec{g_2}, \{g_{3,i}\}_{i=0}^k\big)$ as specified in the second paragraph of the proof of Theorem 1. It then generates the rest of pms and mpk as follows. The algorithm $\mathcal{B}$ defines an arbitrary encoding

ς of the attributes with the restriction that the encodings of the attributes in $\mathcal{P}\backslash S^\star$ correspond to the opposite of the roots of $f(X)$, while the encodings of the elements in $S^\star$ correspond to the opposite of some of the roots of $g(X)$.

The values $\{d_1, \ldots, d_{n+t^\star-1-s^\star}\}$ corresponding to the first $n+t^\star-1-s^\star$ dummy attributes are defined as the opposite of the rest of the roots of $g(X)$ (in some arbitrary order). For $j = n+t^\star-s^\star, \ldots, n-1$, the d_j's can be chosen at random in $\mathbb{Z}_p$ until they are distinct from $\{x_1, \ldots, x_{2n+t^\star-1-s^\star}, d_{n+t^\star-s^\star}, \ldots, d_{j-1}\}$.

The algorithm $\mathcal{B}$ defines $g := g_0^{f(\gamma)}$. Note that $\mathcal{B}$ can compute g with the elements of line (l.1) of its input, since f is a polynomial of degree $\tilde{\ell} = n - s^\star$. To complete the setup phase, $\mathcal{B}$ sets $h = h_0$ and computes

- $u = g^{\alpha\gamma} = g_0^{\alpha\cdot\gamma\cdot f(\gamma)}$ with line (l.3) of its input, which is possible since $Xf(X)$ is a polynomial of degree $\tilde{\ell}+1$. Indeed, $\alpha\cdot\gamma\cdot f(\gamma)$ is a linear combination of $\{\alpha\gamma, \ldots, \alpha\gamma^{\tilde{\ell}+1}\}$ and the coefficients of this linear combination are known to $\mathcal{B}$, so the value u can be computed from line (l.3).
- $v = e(g,h)^\alpha = e(g_0^{f(\gamma)\alpha}, h_0)$ with line (l.3) for $g_0^{f(\gamma)\alpha}$.
- $\{h^{\alpha\gamma^i}\}_{i=0,\ldots,2n-1}$ from line (l.6) of its input.

Private key queries: Whenever the adversary $\mathcal{F}$ makes a key extraction query for a subset of attributes $\Omega \subset \mathcal{P}$ satisfying that $0 \leq |\Omega_S = \Omega \cap S^\star| \leq t^\star - 1$, the algorithm $\mathcal{B}$ must produce a tuple of the form

$$\mathsf{sk}_\Omega = \Big(\Big\{ g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}} \Big\}_{\mathsf{at}\in\Omega}, \Big\{ h^{r\gamma^i} \Big\}_{i=0,\ldots,n-2}, h^{\frac{r-1}{\gamma}} \Big),$$

for some random value $r \in \mathbb{Z}_p$. To do so, $\mathcal{B}$ implicitly defines $r = (\omega y_\Omega \gamma + 1)Q_\Omega(\gamma)$, where y_Ω is randomly picked in $\mathbb{Z}_p$, and the polynomial $Q_\omega(X)$ is defined as $Q_\Omega(\gamma) = 1$ when $|\Omega_S| = 0$, or $Q_\Omega(X) = \lambda_\Omega \cdot \prod_{\mathsf{at}\in\Omega_S} (X + \varsigma(\mathsf{at}))$ otherwise, in which case $\lambda_\Omega = (\prod_{\mathsf{at}\in\Omega_S} \varsigma(\mathsf{at}))^{-1}$.

The elements which form sk_Ω are then computed as follows:

- For any attribute $\mathsf{at} \in \Omega_S$, $\mathcal{B}$ defines

$$Q_{\mathsf{at}}(\gamma) = Q_\Omega(\gamma)/(\gamma + \varsigma(\mathsf{at})) = \lambda_\Omega \cdot \prod_{\tilde{\mathsf{at}}\in\Omega_S,\ \tilde{\mathsf{at}}\neq\mathsf{at}} (\gamma + \varsigma(\tilde{\mathsf{at}}))$$

  and then $g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}} = g_0^{f(\gamma)\omega y_\Omega \gamma Q_{\mathsf{at}}(\gamma)} \cdot g_0^{f(\gamma)Q_{\mathsf{at}}(\gamma)}$. The first factor of the product (whose exponent is a polynomial in γ of degree at most $(n-s^\star)+1+t^\star-2$) can be computed from line (l.2), whereas the second factor (whose exponent is a polynomial in γ of degree at most $(n-s^\star)+t^\star-2$) can be computed from line (l.1).
- For any attribute $\mathsf{at} \in \Omega \setminus \Omega_S$, $\mathcal{B}$ defines the polynomial $f_{\mathsf{at}}(X) = f(X)/(X+\varsigma(\mathsf{at}))$ and then $g^{\frac{r}{\gamma+\varsigma(\mathsf{at})}} = g_0^{f_{\mathsf{at}}(\gamma)\omega y_\Omega \gamma Q_\Omega(\gamma)} \cdot g_0^{f_{\mathsf{at}}(\gamma)Q_\Omega(\gamma)}$. Again, the first factor of this product can be computed from line (l.2), and the second factor can be computed from line (l.1).
- The values $\Big\{ h^{r\gamma^i} \Big\}_{i=0,\ldots,n-2}$ can be computed from line (l.4) and (l.5), since $h^{r\gamma^i} = h^{Q_\Omega(\gamma)\omega y_\Omega \gamma^{i+1}} \cdot h^{Q_\Omega(\gamma)\gamma^i}$.
- Finally, $\mathcal{B}$ has to compute $h^{\frac{r-1}{\gamma}} = h^{Q_\Omega(\gamma)\omega y_\Omega} \cdot h^{\frac{Q_\Omega(\gamma)-1}{\gamma}}$. The first factor of the product can be computed from line (l.5) and the second factor can be computed from line (l.4), since by definition of λ_Ω, $Q_\Omega(X)$ is a polynomial with independent term equal to 1 and thus $\frac{Q_\Omega(\gamma)-1}{\gamma}$ is a linear combination of $\{1, \gamma, \ldots, \gamma^{t-2}\}$.

Note that $Q_\Omega(\gamma) \neq 0$ (otherwise $\gamma = \varsigma(\mathsf{at})$ for some $\mathsf{at} \in \Omega_S$ and γ is public), in which case it is not hard to see that r is uniformly distributed in $\mathbb{Z}_p$. If the choice of y_Ω leads to $r = 0$ (which occurs only with negligible probability anyhow), it suffices to pick a different value for y_Ω. That is, in the simulation r is uniformly distributed in $\mathbb{Z}_p$.

Signing queries: At each signing query, $\mathcal{F}$ chooses a message Msg and a threshold access policy $\Gamma = (t, S)$, where S is an attribute set of size $s \leq n$ and $t \in \{1, \ldots, s\}$. To answer such a query, $\mathcal{B}$ computes $M = H(\mathsf{Msg}, \Gamma) \in \{0,1\}^k$ and parses it as a k-bit string $M = m_1 \ldots m_k \in \{0,1\}^k$. It evaluates the functions

$$J(M) = -\nu \cdot \tau + \rho_0 + \sum_{j=1}^{k} \rho_j \cdot m_j, \tag{14}$$

$$K_1(M) = \xi_{0,1} + \sum_{j=1}^{k} \xi_{j,1} \cdot m_j, \qquad K_2(M) = \xi_{0,2} + \sum_{j=1}^{k} \xi_{j,2} \cdot m_j, \tag{15}$$

for which the message-dependent vector $\vec{g}_{3,M}$ equals $\vec{g}_{3,M} = \vec{g_1}^{K_1(M)} \cdot \vec{g_2}^{K_2(M)} \cdot (1,1,g)^{-J(M)}$ and aborts in the event that $J(M) = 0$. Otherwise, $\mathcal{B}$ can generate a signature $\sigma = \big(\vec{C}_{T_1}, \vec{C}_{T_2}, \vec{C}_\theta, \vec{\pi}_1, \vec{\pi_2}\big)$ by simulating NIZK proofs for relations (2)-(3) as follows. First, $\mathcal{B}$ computes the query-dependent vector $\vec{g}_{3,M} = \vec{g}_{3,0} \cdot \prod_{i=1}^{k} \vec{g_{3,i}}^{m_i} = \vec{g_1}^{K_1(M)} \cdot \vec{g_2}^{K_2(M)} \cdot (1,1,g)^{-J(M)}$. Observe that, as long as $J(M) \neq 0$, $\mathbf{g}_M = (\vec{g}_1, \vec{g}_2, \vec{g}_{3,M})$ forms a Groth-Sahai CRS for the WI setting (meaning that commitments generated using it are always perfectly hiding) and using the trapdoor information $(K_1(M), K_2(M), J(M))$, $\mathcal{B}$ can simulate proofs without knowing witnesses.

To this end, $\mathcal{B}$ generates $\vec{C}_{T_1}$, $\vec{C}_{T_2}$ and $\vec{C}_\theta$ as commitments to the identity element $1_\mathbb{G}$ and uses the variable assignment $T_1 = T_2 = \Theta = 1_\mathbb{G}$ to honestly generate a proof for equation (2). As for relation (3), $\mathcal{B}$ uses the trapdoor $(K_1(M), K_2(M), J(M))$ of the simulated reference string $\mathbf{g}_M$ to generate a fake proof that $\Theta = h$. Namely, with $\vec{C}_\theta = \vec{g_1}^{r_\theta} \cdot \vec{g_2}^{s_\theta} \cdot \vec{g}_{3,M}^{t_\theta}$ being a commitment to $1_\mathbb{G}$, the proof elements

$$\pi_{2,1} = g^{r_\theta} \cdot h^{-K_1(M)/J(M)}, \qquad \pi_{2,2} = g^{s_\theta} \cdot h^{-K_2(M)/J(M)}, \qquad \pi_{2,3} = g^{t_\theta} \cdot h^{1/J(M)},$$

are easily seen to satisfy the verification equation (5). To make sure that $\vec{\pi_2} = (\pi_{2,1}, \pi_{2,2}, \pi_{2,3})$ is uniformly distributed in the space of valid proofs, $\mathcal{B}$ then performs a proof re-randomization as explained in [17].

Forgery: at the end of the game, the adversary $\mathcal{F}$ outputs $\sigma^\star = \big(\vec{C}^\star_{T_1}, \vec{C}^\star_{T_2}, \vec{C}^\star_\theta, \vec{\pi}^\star_1, \vec{\pi_2}^\star\big)$ for some message $\mathsf{Msg}^\star$ and the expected target access policy $\Gamma^\star = (t^\star, S^\star)$. At this step, $\mathcal{B}$ computes $M^\star = m^\star_1 \ldots m^\star_k = H(\mathsf{Msg}^\star, \Gamma^\star)$ and evaluates the functions $J(M^\star)$, $K_1(M^\star)$ and $K_2(M^\star)$ as per (14)-(15). It aborts if it holds that either:

1. The hash value $M^\star = H(\mathsf{Msg}^\star, \Gamma^\star)$ is such that $J(M^\star) \neq 0$.
2. $\mathcal{F}$ made a signing query (Msg, Γ) for which $(\mathsf{Msg}, \Gamma) \neq (\mathsf{Msg}^\star, \Gamma^\star)$ and $H(\mathsf{Msg}, \Gamma) = H(\mathsf{Msg}^\star, \Gamma^\star)$.

Situation 2 obviously breaks the collision-resistance of H and we will only have to bound the probability of situation 1 occurring. If neither situation occurs (which implies $J(M^\star) = 0$), the proofs $\vec{\pi}^\star_1, \vec{\pi_2}^\star$ must be valid w.r.t. the Groth-Sahai CRS $\mathbf{g}_{M^\star} = (\vec{g}_1, \vec{g}_2, \vec{g}_{3,M^\star})$, for which the third vector is given by

$$\vec{g}_{3,M^\star} = \vec{g}_{3,0} \cdot \prod_{i=1}^{k} \vec{g}_{3,i}^{\,m^\star_i} = \vec{g_1}^{K_1(M^\star)} \cdot \vec{g_2}^{K_2(M^\star)} = \big(g_1^{K_1(M^\star)}, g_2^{K_2(M^\star)}, g^{K_1(M^\star)+K_2(M^\star)}\big),$$

so that $\mathbf{g}_{M^\star}$ forms a perfectly sound Groth-Sahai CRS. The latter perfect soundness property guarantees that, with respect to $\mathbf{g}_{M^\star}$, $\vec{C}^\star_{T_1}$ and $\vec{C}^\star_{T_2}$ are *extractable* Groth-Sahai commitments and that, by BBS-decrypting them using $a, b \in \mathbb{Z}_p$, $\mathcal{B}$ obtains values T_1, T_2 that satisfy relation (1). From the values T_1, T_2, $\mathcal{B}$ can easily find the solution to the $(n - s^\star, n + t^\star - 1, t^\star + 1)$-aMSE-CDH problem. Indeed, from $h_0^{\kappa \cdot g(\gamma)}$ (line (l.4)) and $\left(g_0^{\kappa \cdot \gamma f(\gamma)}\right)^{-1}$ (and line (l.1)), algorithm $\mathcal{B}$ simply computes

$$e(T_1, h_0^{\kappa \cdot g(\gamma)}) e(T_2, \left(g_0^{\kappa \cdot \gamma f(\gamma)}\right)^{-1}) = e(g_0, h_0)^{\kappa f(\gamma)}.$$

Using a similar analysis to the probabilistic analysis of the simulator in Waters' identity-based encryption scheme [32] (but without the artificial abort step), we find that, with probability $1/(8 \cdot q \cdot (k+1))$, $\mathcal{B}$ has $J(M^\star) = 0$ in the forgery stage whereas $J(M) \neq 0$ in all signing queries. Hence, if $\mathcal{F}$'s advantage is ε, $\mathcal{B}$'s probability of success is $\varepsilon/(8 \cdot q \cdot (k+1))$, which is non-negligible. □

A.2 Proof of Theorem 3

We show that a forger $\mathcal{F}$ allows constructing either a collision-finder for H or an algorithm $\mathcal{B}$ that computes $g^{\gamma^{N+1}}$ from $(g, g^\gamma, \ldots, g^{\gamma^N}, g^{\gamma^{N+2}}, \ldots, g^{\gamma^{2N}})$, where $N = 2n + 1$. In the following notations, we denote by $\vec{\gamma}$ the vector $\vec{\gamma} = (\gamma, \gamma^2, \ldots, \gamma^N)$ and also define $z_i = g^{\gamma^i}$, for each $i \in \{1, \ldots, 2N\} \backslash \{N+1\}$.

At the very beginning of the attack game, the forger $\mathcal{F}$ declares which policy $\Gamma^\star = (t^\star, S^\star)$ it wants to be challenged upon, where $t^\star \in \{1, \ldots, |S^\star|\}$. Armed with this information, $\mathcal{B}$ prepares the trusted public parameters pms and the master public key mpk as follows. It first selects a set $\mathcal{D}$ of n dummy attributes, then $\mathcal{B}$ computes the vector $\vec{Y}$ associated to the polynomial $P_{S^\star}(Z)$ according to equation (8) using a set $\mathcal{D}_{n-t^\star}$ containing the first $n - t^\star$ dummy attributes. More precisely, $\mathcal{B}$ chooses θ_0, δ_0 as well as a random vector $\vec{\theta} \xleftarrow{R} \mathbb{Z}_p^N$ and computes $\vec{H} = (h_1, \ldots, h_N)^\top = g^{\vec{\gamma}} \cdot g^{\vec{\theta}}$ (which implicitly sets $\vec{\alpha} = \vec{\gamma} + \vec{\theta}$), $h_0 = g^{\theta_0} \cdot g^{-\langle \vec{\gamma}, \vec{Y} \rangle}$ and $e(g,g)^\alpha = e(z_1, z_N)^{\delta_0}$, so that the corresponding (unknown) master secret is $g^\alpha = z_{N+1}^{\delta_0}$.

In addition, $\mathcal{B}$ prepares a vector $\vec{U} = (u_0, \ldots, u_k)^\top \in \mathbb{G}^{k+1}$ that will serve as a key for a programmable hash function [19], as in Waters' identity-based encryption scheme [32]. Namely, $\mathcal{B}$ randomly picks $\nu \xleftarrow{R} \{0, \ldots, k\}$, $\phi_0, \phi_1, \ldots, \phi_k \xleftarrow{R} \mathbb{Z}_p$ and $\rho_0, \rho_1, \ldots, \rho_k \xleftarrow{R} \{0, \ldots, \tau - 1\}$, where $\tau = 2q$ (here q is an upper bound on the number of signing queries) and defines $u_i = z_1^{\rho_i} \cdot g^{\phi_i}$ for $i = 1, \ldots, k$ and $u_0 = z_1^{-\nu \cdot \tau + \rho_0} \cdot g^{\phi_0}$. Finally, $\mathcal{B}$ selects a collision-resistant hash function H. The master public key $\mathsf{mpk} = \left(g, e(g,g)^\alpha, h_0, \vec{H}, \vec{U}, \mathcal{D}, H\right)$ is given to $\mathcal{F}$.

In the following, for any $\omega \in \mathbb{Z}_p$, we define the vector $\vec{X}_\omega^n = (1, \omega, \ldots, \omega^{n-1})^\top$. We note that given any set $S \subset \mathbb{Z}_p$ of cardinality less than n, the vectors $\{\vec{X}_\omega^n\}_{\omega \in S}$ are linearly independent.

Private key queries: During the game, $\mathcal{F}$ can obtain private keys for any attribute set Ω such that $|\Omega \cap S^\star| < t^\star$.

Since $|S^\star \cap \Omega| < t^\star$, the cardinality of $(S^\star \cap \Omega) \cup \mathcal{D}_{n-t^\star}$ is strictly less than n. Consequently, the vector $\vec{X}_0^n = (1, 0, \ldots, 0)^\top$ cannot be in the span of the vectors $\{\vec{X}_\omega^n\}_{\omega \in (S^\star \cap \Omega) \cup \mathcal{D}_{n-t^\star}}$. Hence, there must exist an efficiently computable vector $\vec{\tau}$ such that $\langle \vec{X}_\omega^n, \vec{\tau} \rangle = 0$ for any $\omega \in (S^\star \cap \Omega) \cup \mathcal{D}_{n-t^\star}$ and such that $\langle \vec{X}_0^n, \vec{\tau} \rangle \neq 0$ (according to Proposition 1 in [16]). Let μ denote this value $\langle \vec{X}_0^n, \vec{\tau} \rangle$.

To construct a private key, $\mathcal{B}$ has to define a vector $\vec{u} = \alpha \cdot \vec{\beta} \in \mathbb{Z}_p^n$ satisfying the constraint $\langle \vec{X}_0^n, \vec{u} \rangle = \alpha$, for some random constrained vector $\vec{\beta} = (1, \tilde{\beta}_1, \ldots, \tilde{\beta}_{n-1})^\top$ that implicitly defines the coefficients of $Q_\Omega[X]$ as $\beta_i = \alpha \cdot \tilde{\beta}_i$ for $i = 1, \ldots, n-1$. To this end, $\mathcal{B}$ proceeds as in the proof

of Theorem 3 in [16], by implicitly setting $\vec{u}$ as $\vec{u} = \vec{v} + \psi \cdot \vec{\tau}$, where $\vec{v} = (v_1, \ldots, v_n)^\top \in \mathbb{Z}_p^n$ is a randomly chosen vector and $\psi = (\alpha - v_1)/\mu$, so that $\langle \vec{X}_0^n, \vec{u} \rangle = \alpha$.

The task of $\mathcal{B}$ is thus to simulate (without knowing $\vec{u}$) the private key components for any $\omega \in \Omega \cup \mathcal{D}$, which are:

$$\mathsf{SK}_\omega = \big(D_{\omega,1}, D_{\omega,2}, \{K_{\omega,i}\}_{i=1}^{N-1}\big) = \big(g^{Q_\Omega(\omega)} \cdot h_0^{r_\omega}, g^{r_\omega}, \{h_1^{-\omega^i} h_{i+1}\}_{i=1}^{N-1}\big),$$

where $Q_\Omega(\omega) = \langle \vec{X}_\omega^n, \vec{u} \rangle$.

Let us divide the set $\Omega \cup \mathcal{D}$ into two subsets: $\Upsilon_1 = (\Omega \cup \mathcal{D}) \cap ((S^\star \cap \Omega) \cup \mathcal{D}_{n-t^\star})$ and Υ_2, its complement.

1. For each $\omega \in \Upsilon_1$, we have $Q_\Omega(\omega) = \langle \vec{X}_\omega^n, \vec{u} \rangle = \langle \vec{X}_\omega^n, \vec{v} \rangle$ which is efficiently computable by $\mathcal{B}$. Hence, $\mathcal{B}$ can simply pick $r_\omega \xleftarrow{R} \mathbb{Z}_p^*$ and define

$$D_\omega = \big(D_{\omega,1}, D_{\omega,2}, \{K_{\omega,i}\}_{i=1}^{N-1}\big) = \Big(g^{Q_\Omega(\omega)} \cdot h_0^{r_\omega},\ g^{r_\omega},\ \{(h_1^{-\omega^i} h_{i+1})^{r_\omega}\}_{i=1}^{N-1}\Big).$$

2. For each $\omega \in \Upsilon_2$, $\mathcal{B}$ can construct a valid key component $\mathsf{SK}_\omega = \big(D_{\omega,1}, D_{\omega,2}, \{K_{\omega,i}\}_{i=1}^{N-1}\big)$ in two steps. The first step consists in building a tuple of the form

$$\big(D_{\omega,1}^*, D_{\omega,2}^*, \{K_{\omega,i}^*\}_{i=1}^{N-1}\big) = \Big(g^\alpha \cdot h_0^{\tilde{r_\omega}},\ g^{\tilde{r_\omega}},\ \{(h_1^{-\omega^i} h_{i+1})^{\tilde{r_\omega}}\}_{i=1}^{N-1}\Big)$$

using that attribute ω is not in the set $(S^\star \cup \mathcal{D}_{n-t^\star})$. To this end, $\mathcal{B}$ proceeds as in [9]. Let M_ω be the $N \times (N-1)$ matrix

$$M_\omega = \begin{pmatrix} -\omega & & & \\ \omega^2 & & \cdots & \\ \cdots & & & \\ -\omega^{N-1} & & & \end{pmatrix} \begin{matrix} \\ \\ I_{N-1} \\ \\ \end{matrix}.$$

Pick $\xi_1 \xleftarrow{R} \mathbb{Z}_p^*$. Note that the vector $\vec{\xi} = \xi_1 \cdot (1, \omega, \ldots, \omega^{N-1})^\top$ satisfies that $\vec{\xi}^{\,\top} M_\omega = \vec{0}$ while $\langle \vec{Y}, \vec{\xi} \rangle = \xi_1 \cdot P_{S^\star}(\omega) \neq 0$.
The simulator $\mathcal{B}$ computes

$$\big(D_{\omega,1}^*, D_{\omega,2}^*\big) = \Big(g^\alpha \cdot h_0^{\tilde{r}_\omega},\ g^{\tilde{r}_\omega}\Big) \tag{16}$$

and

$$\big(K_{\omega,1}^*, \ldots, K_{\omega,N-1}^*\big)^\top = g^{\tilde{r}_\omega M_\omega^\top \vec{\alpha}}, \tag{17}$$

with $\vec{\alpha} = (\alpha_1, \ldots, \alpha_N)^\top$ and where the exponent $\tilde{r}_\omega$ is defined as

$$\tilde{r}_\omega = r + \delta_0 \langle (\gamma^N, \gamma^{N-1}, \ldots, \gamma)^\top, \vec{\xi} \rangle / \langle \vec{Y}, \vec{\xi} \rangle$$

for some $r \xleftarrow{R} \mathbb{Z}_p$ chosen by $\mathcal{B}$. Since $g^{M_\omega^\top \vec{\alpha}} = (h_1^{-\omega} h_2, \ldots, h_1^{-\omega^{N-1}} h_N)^\top$, if we can argue that both expressions (16) and (17) are computable by $\mathcal{B}$, we will have concluded the first step of the key generation process.

Observe that for any vector $\vec{f} \in \mathbb{Z}_p^N$ the coefficient of γ^{N+1} in the product $\tilde{r}_\omega \langle \vec{f}, \vec{\gamma} \rangle$ is $\delta_0 \langle \vec{f}, \vec{\xi} \rangle / \langle \vec{Y}, \vec{\xi} \rangle$. The reason why $\mathcal{B}$ can compute the second factor of side of $D_{\omega,1}^*$ in expression (16) is that the coefficient of $g^{\gamma^{N+1}}$ in $D_{\omega,1}^*$ is 0. Indeed, $D_{\omega,1}^* = g^\alpha \cdot h_0^{\tilde{r}_\omega} = z_{N+1}^{\delta_0} \cdot \big(g^{\theta_0} \cdot g^{-\langle \vec{\gamma}, \vec{Y} \rangle}\big)^{\tilde{r}_\omega}$ and the

coefficient of γ^{N+1} is $-\delta_0$ in the product $-\tilde{r}_j\langle\vec{\gamma},\vec{Y}\rangle$, as we can see applying the observation above in the case $\vec{f}=\vec{Y}$.

Since $M_\omega{}^\top\vec{\xi}=\vec{0}$, applying again the observation above, now to the case where $\vec{f}^\top$ is successively set as the rows of $M_\omega^\top$, we have that the unknown term $z_{N+1}=g^{\gamma^{N+1}}$ does not appear in $g^{\tilde{r}_j M_\omega^\top\vec{\alpha}}$, which is thus computable.

This concludes the first step of the key generation process. In the second step, we just have to turn $\big(D^*_{\omega,1},D^*_{\omega,2},\{K^*_{\omega,i}\}_{i=1}^{N-1}\big)$ into a suitable key component SK_ω. Note that

$$\langle\vec{X}^n_\omega,\vec{u}\rangle=\langle\vec{X}^n_\omega,\vec{v}\rangle+\psi\cdot\langle\vec{X}^n_\omega,\vec{\tau}\rangle=\sum_{j=1}^{n}\omega^{j-1}\Big(v_j+\frac{(\alpha-v_1)}{\mu}\cdot\tau_j\Big)=\kappa_1\cdot\alpha+\kappa_2,$$

where the coefficients $\kappa_1=(\sum_{j=1}^n\omega^{j-1}\tau_j)\cdot\mu^{-1}$ and $\kappa_2=\mu^{-1}\cdot\sum_{j=1}^n\omega^{j-1}\big(\mu v_j-v_1\tau_j\big)$ are both computable, so that $\mathcal{B}$ can obtain a well-formed tuple $\mathsf{SK}_\omega=(D_{\omega,1},D_{\omega,2},\{K_{\omega,i}\}_{i=1}^{N-1})$ by setting

$$\mathsf{SK}_\omega=(D_{\omega,1},D_{\omega,2},\{K_{\omega,i}\}_{i=1}^{N-1})=\Big(D^*_{\omega,1}{}^{\kappa_1}\cdot g^{\kappa_2}\cdot h_0^{r'_\omega},D^*_{\omega,2}{}^{\kappa_1}\cdot g^{r'_\omega},\{K^*_{\omega,i}{}^{\kappa_1}\cdot(h_1^{-\omega^i}\cdot h_{i+1})^{r'_\omega}\}_{i=1}^{N-1}\Big),$$

for a randomly chosen $r'_j\xleftarrow{R}\mathbb{Z}_p$.

Signing queries: At any time, $\mathcal{F}$ is also allowed to obtain signatures on arbitrary messages. At each signing query, $\mathcal{F}$ supplies a message Msg and a threshold access policy $\Gamma=(t,S)$, where S is an attribute set of size $s\leq n$ and $t\in\{1,\ldots,s\}$. To answer such a query, $\mathcal{B}$ computes $M=H(\mathsf{Msg},\Gamma)\in\{0,1\}^k$ and parses it as a k-bit string $m_1\ldots m_k$. It evaluates the functions

$$J(M)=-\nu\cdot\tau+\rho_0+\sum_{j=1}^{k}\rho_j\cdot m_j\qquad\text{and}\qquad K(M)=\phi_0+\sum_{j=1}^{k}\phi_j\cdot m_j,\tag{18}$$

for which the programmable hash function implemented by the vector $\vec{U}=(u_0,\ldots,u_k)^\top\in\mathbb{G}^{k+1}$ is such that $u_0\cdot\prod_{j=1}^k u_j^{m_j}=z_1^{J(M)}\cdot g^{K(M)}$, and aborts in the event that $J(M)=0$. Then, $\mathcal{B}$ constructs the vector $\vec{Y}=(y_1,\ldots,y_N)^\top$ whose coordinates are the coefficients of the polynomial $P_S(Z)$ which is obtained following relation (8), by augmenting S with $n-t$ dummy attributes. Recall that $\mathcal{B}$ has to generate a signature of the form

$$\sigma_1=g^\alpha\cdot\big(h_0\cdot\prod_{i=1}^{N}h_i^{y_i}\big)^r\cdot\big(u_0\cdot\prod_{j=1}^{k}u_j^{m_j}\big)^{\tilde{z}},\qquad\sigma_2=g^r,\qquad\sigma_3=g^{\tilde{z}},\tag{19}$$

for some $r,\tilde{z}\in_R\mathbb{Z}_p$. To this end, $\mathcal{B}$ uses the usual technique (which dates back to [3]) consisting in implicitly defining $\tilde{z}=z-\frac{\gamma^N\cdot\delta_0}{J(M)}$, for a randomly chosen $z\xleftarrow{R}\mathbb{Z}_p$, and computing

$$\sigma_1=\big(u_0\cdot\prod_{j=1}^{k}u_j^{m_j}\big)^z\cdot z_N^{-K(M)\cdot\delta_0/J(M)}\cdot\big(h_0\cdot\prod_{i=1}^{n}h_i^{y_i}\big)^r,\qquad\sigma_2=g^r,\qquad\sigma_3=g^z\cdot z_N^{-\delta_0/J(M)},$$

for a random $r\xleftarrow{R}\mathbb{Z}_p$. Since $\tilde{z}=z-\frac{\gamma^N\cdot\delta_0}{J(M)}$ and α is implicitly defined as $\alpha=\gamma^{N+1}\cdot\delta_0$, the above triple is easily seen to have the required distribution (19).

Forgery: The game ends with the adversary outputting a forgery $\sigma^\star = (\sigma_1^\star, \sigma_2^\star, \sigma_3^\star)$ for some message $\mathsf{Msg}^\star$ and the target access policy $\Gamma^\star = (t^\star, S^\star)$ without having made any illegal query. At this step, $\mathcal{B}$ computes $m_1^\star \ldots m_k^\star = M^\star = H(\mathsf{Msg}^\star, \Gamma^\star)$ and evaluates the functions $J(M^\star)$ and $K(M^\star)$ as per (18). It aborts if it holds that either:

1. The hash value $M^\star = H(\mathsf{Msg}^\star, \Gamma^\star)$ is such that $J(M^\star) \neq 0$.
2. $\mathcal{F}$ made a signing query (Msg, Γ) such that $(\mathsf{Msg}, \Gamma) \neq (\mathsf{Msg}^\star, \Gamma^\star)$ and $H(\mathsf{Msg}, \Gamma) = H(\mathsf{Msg}^\star, \Gamma^\star)$.

We will see that situation 1 is avoided with noticeable probability and case 2 obviously breaks the collision-resistance of H. If neither of the above situation occurs (and thus $J(M^\star) = 0$), $\mathcal{B}$ can obtain the searched value $z_{N+1} = g^{\gamma^{N+1}}$ as follows. Since the vector $\vec{Y} = (y_1, \ldots, y_N)^\top$ derived from $\Gamma^\star$ is such that $h_0 \cdot \prod_{i=1}^N h_i^{y_i} = g^{\theta_0 + \langle \vec{\theta}, \vec{Y} \rangle}$ and given that $J(M^\star) = 0$ and $\sigma^\star$ satisfies the verification equation (13), we must have

$$e(g,g)^{(\gamma^{N+1}) \cdot \delta_0} = \frac{e(\sigma_1^\star, g)}{e(\sigma_2^\star, g^{\theta_0 + \langle \vec{\theta}, \vec{Y} \rangle}) \cdot e(\sigma_3^\star, g^{K(M^\star)})},$$

which implies that $z_{N+1} = \left(\frac{\sigma_1^\star}{\sigma_2^{\star\, \theta_0 + \langle \vec{\theta}, \vec{Y} \rangle} \cdot \sigma_3^{\star\, K(M^\star)}} \right)^{1/\delta_0}$ is computable by $\mathcal{B}$.

The same analysis as in Waters' identity-based encryption scheme [32] shows that, with probability $1/(8 \cdot q \cdot (k+1))$, $\mathcal{B}$ gets lucky and has $J(M^\star) = 0$ in the forgery stage whereas $J(M) \neq 0$ in all signing queries. It comes that, if $\mathcal{F}$'s advantage is ε, $\mathcal{B}$'s success probability is $\varepsilon/(8 \cdot q \cdot (k+1))$, which is non-negligible. □