Efficient Attribute-Based Signatures for Non-Monotone Predicates in the Standard Model

Tatsuaki Okamoto[1] and Katsuyuki Takashima[2]

[1] NTT, 3-9-11 Midori-cho, Musashino-shi, Tokyo, 180-8585 Japan
okamoto.tatsuaki@lab.ntt.co.jp
[2] Mitsubishi Electric, 5-1-1, Ofuna, Kamakura, Kanagawa, 247-8501 Japan
Takashima.Katsuyuki@aj.MitsubishiElectric.co.jp

Abstract. This paper presents a *fully* secure (*adaptive*-predicate unforgeable and private) attribute-based signature (ABS) scheme in the *standard* model. The security of the proposed ABS scheme is proven under standard assumptions, the decisional linear (DLIN) assumption and the existence of collision resistant (CR) hash functions. The admissible predicates of the proposed ABS scheme are more general than those of the existing ABS schemes, i.e., the proposed ABS scheme is the first to support general *non-monotone* predicates, which can be expressed using *NOT* gates as well as AND, OR, and Threshold gates, while the existing ABS schemes only support *monotone* predicates. The proposed ABS scheme is efficient and practical. Its efficiency is comparable to (several times worse than) that of the most efficient (almost optimally efficient) ABS scheme the security for which is proven in the generic group model.

1 Introduction

1.1 Background

The concept of digital signatures was introduced in the seminal paper by Diffie and Hellman in 1976. In this concept, a pair comprising a secret signing key, sk, and public verification key, pk, is generated for a signer, and signature σ of message m generated using sk is verified by the corresponding pk. Hence, the signer of (m, σ) using sk is identified through pk. Although it is one of the requirements of signatures, there is no flexibility or privacy in the relationship between signers and claims attested by signatures due to the tight relation between sk and pk.

Recently, versatile and privacy-enhanced variants of digital signatures have been studied, where the relation between a signing key and verification key is more flexible or sophisticated. In this class of signatures, the signing key and verification key are parameterized by *attribute* $\boldsymbol{x}$ and *predicate* $\boldsymbol{v}$, respectively, and signed message (m, σ) generated by the signing key with parameter $\boldsymbol{x}$, $\mathsf{sk}_{\boldsymbol{x}}$, is correctly verified by public-key pk and parameter $\boldsymbol{v}$, $(\mathsf{pk}, \boldsymbol{v})$, iff predicate $\boldsymbol{v}$ accepts attribute $\boldsymbol{x}$, i.e., $\boldsymbol{v}(\boldsymbol{x})$ holds. The privacy of signers in this class of signatures requires that a signature (for predicate $\boldsymbol{v}$) generated by $\mathsf{sk}_{\boldsymbol{x}}$ (where $\boldsymbol{v}(\boldsymbol{x})$ holds) release no information regarding attribute $\boldsymbol{x}$ except that $\boldsymbol{v}(\boldsymbol{x})$ holds.

When predicate $\boldsymbol{v}$ is the equality with parameter v (i.e., $\boldsymbol{v}(x)$ holds iff $x = v$), the class of signatures for this predicate is *identity-based signatures* (IBS) [25]. Here note that there is no room for privacy in IBS, since predicate $\boldsymbol{v}$ uniquely identifies attribute x of the signer's secret key, sk_x, such that $x = v$.

Group signatures [9] are also in this class of signatures with another type of predicate $\boldsymbol{v}$, where $\boldsymbol{v}(\boldsymbol{x})$ holds iff predicate parameter v is the group identity (or pk_v is a public key identifying group v) and attribute x is a member identity of group v (or sk_x is a secret key of member x of group v). Due to the privacy requirement, signatures generated using sk_x release no information regarding member identity x except that x is a member of group v (Note that the concept of group signatures traditionally requires the *privacy-revocation* property as well as the above-mentioned privacy).

Recently, this class of signatures with more sophisticated predicates, *attribute-based signatures* (ABS), has been extensively studied [11–13, 16–19, 24, 27], where $\boldsymbol{x}$ for signing key $\mathsf{sk}_{\boldsymbol{x}}$ is a tuple of attributes $(x_1, \ldots, x_i)$, and $\boldsymbol{v}$ for verification is a threshold or access structure predicate. The widest class of predicates in the existing ABS schemes are monotone access structures [18, 19], where predicate $\boldsymbol{v}$ is specified by a monotone span program (MSP), (M, ρ), along with a tuple of attributes $(v_1, \ldots, v_j)$, and $\boldsymbol{v}(\boldsymbol{x})$ holds iff MSP (M, ρ) accepts the truth-value vector of $(\mathsf{T}(x_{i_1} = v_1), \ldots, \mathsf{T}(x_{i_j} = v_j))$. Here, $\mathsf{T}(\psi) := 1$ if ψ is true, and $\mathsf{T}(\psi) := 0$ if ψ is false (For example, $\mathsf{T}(x = v) := 1$ if $x = v$, and $\mathsf{T}(x = v) := 0$ if $x \neq v$). In general, such a predicate can be expressed using AND, OR, and Threshold gates.

An example of such monotone predicate $\boldsymbol{v}$ for ABS is (Institute = Univ. A) AND (TH2((Department = Biology), (Gender = Female), (Age = 50's)) OR (Position = Professor)), where TH2 means the threshold gate with threshold value 2. Attribute $\boldsymbol{x}_A$ of Alice is ((Institute := Univ. A), (Department := Biology), (Position := Postdoc), (Age := 30), (Gender := Female))), and attribute $\boldsymbol{x}_B$ of Bob is ((Institute := Univ. A), (Department := Mathematics), (Position := Professor), (Age := 45) (Gender := Male))). Although their attributes, $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$, are quite different, it is clear that $\boldsymbol{v}(\boldsymbol{x}_A)$ and $\boldsymbol{v}(\boldsymbol{x}_B)$ hold, and that there are many other attributes that satisfy $\boldsymbol{v}$. Hence Alice and Bob can generate a signature on this predicate, and due to the privacy requirement of ABS, a signature for $\boldsymbol{v}$ releases no information regarding the attribute or identity of the signer, i.e., Alice or Bob (or other), except that the attribute of the signer satisfies $\boldsymbol{v}$.

There are many applications of ABS such as attribute-based messaging (ABM), attribute-based authentication, trust-negotiation and leaking secrets (see [18, 19] for more details).

The security conditions for ABS are given hereafter (see Section 3.2 for the formal definitions).

Unforgeability: A valid signature should be produced only by a *single* signer whose attribute $\boldsymbol{x}$ satisfies the claimed predicate $\boldsymbol{v}$, not by a collusion of users who pooled their attributes together. More formally, no poly-time adversary can produce a valid signature for a pair comprising predicate and message

$(\boldsymbol{v}, m)$, even if the adversary *adaptively* chooses $(\boldsymbol{v}, m)$ after executing secret-key and signing oracle attacks, provided that $\boldsymbol{x}$ where $\boldsymbol{v}(\boldsymbol{x})$ holds is not queried to the secret-key oracle and $(\boldsymbol{v}, m)$ is not queried to the signing oracle (We simply call this unforgeability "*adaptive*-predicate unforgeability" or more simply "unforgeability").

We can also define a *weaker* class of unforgeability, '*selective*-predicate unforgeability,' where an adversary should choose predicate $\boldsymbol{v}$ for the forgery signature before executing secret-key and signing oracle attacks.

Privacy: A signature for predicate $\boldsymbol{v}$ generated using secret key $\mathsf{sk}_{\boldsymbol{x}}$ releases no information regarding attribute $\boldsymbol{x}$ except that $\boldsymbol{v}(\boldsymbol{x})$ holds.

More formally, for any pair of attributes $(\boldsymbol{x}_1, \boldsymbol{x}_2)$, predicate $\boldsymbol{v}$ and message m, for which $\boldsymbol{v}(\boldsymbol{x}_1)$ and $\boldsymbol{v}(\boldsymbol{x}_2)$ hold simultaneously, the distributions of two valid signatures $\sigma(m, \boldsymbol{v}, \mathsf{sk}_{\boldsymbol{x}_1})$ and $\sigma(m, \boldsymbol{v}, \mathsf{sk}_{\boldsymbol{x}_2})$ are equivalent, where $\sigma(m, \boldsymbol{v}, \mathsf{sk}_{\boldsymbol{x}})$ is a correctly generated signature for $(m, \boldsymbol{v})$ using correct secret key $\mathsf{sk}_{\boldsymbol{x}}$ with attribute $\boldsymbol{x}$ (We simply call this condition "*privacy*").

Full Security: We say that an ABS scheme is *fully* secure if it satisfies *adaptive*-predicate unforgeability and *privacy.*

Maji, Prabhakaran, and Rosulek [18, 19] presented ABS schemes for the widest class of predicates among the existing ABS schemes, monotone access structure predicates, which cover threshold predicates as special cases. The scheme shown in [18] is an almost optimally efficient ABS scheme, but the security was only proven in the generic group model. The scheme shown in [19] is the only existing ABS scheme for which (full) security was proven in the standard model. It is, however, much less efficient and more complicated than the scheme in [18] since it employs the Groth-Sahai NIZK protocols [10] as building blocks.

Li, Au, Susilo, Xie and Ren [16], Li and Kim [17], and Shahandashti and Safavi-Naini [24] presented ABS schemes that are proven to be secure in the standard model. However, the proven security is not the full security, but a weaker level of security with *selective*-predicate unforgeability. Moreover, the admissible predicates in [17] are limited to conjunction or (n, n)-threshold predicates, and those of [16, 24] are limited to (k, n)-threshold predicates.

Guo and Zeng [11] and Yang, Cao and Dong [27] presented ABS schemes for threshold predicates, but their security definitions do not include the *privacy* condition of ABS.

Khader [12, 13] presented ABS schemes for monotone access structure predicates. These schemes, however, do not satisfy the *privacy* condition of ABS, since they only conceal the identity of the signer. They also reveal the attributes that the signer used to generate the signature. In addition, the security is proven in a non-standard model, the random oracle model.

Based on this background, there are two major problems in the existing ABS schemes.

1. No ABS scheme for *non-monotone* predicates, which can be expressed using NOT gates as well as AND, OR and Threshold gates, has been proposed (even in a weaker security notion or a non-standard model).

2. The only fully secure ABS scheme in the *standard* model [19] is much less efficient than the (almost optimally efficient) ABS scheme in the generic group model [18].

Non-monotone predicates should be used in many ABS applications. For example, annual review reports in the Mathematics Department of University A are submitted by reviewers, and these reports are anonymously signed by the reviewers through ABS with some predicates. The predicates may be selected freely by them (signers) except that it should be in the following form: NOT((Institute = Univ. A) AND (Department = Mathematics)) AND $(\cdots)$.

1.2 Our Results

This paper addresses these problems simultaneously.

- This paper proposes the first fully secure (i.e., adaptive-predicate unforgeable and perfectly private) ABS scheme for a wide class of predicates, *non-monotone* access structures, where $\boldsymbol{x}$ for signing key $\mathsf{sk}_{\boldsymbol{x}}$ is a tuple of attributes $(x_1, \ldots, x_i)$, non-monotone predicate $\boldsymbol{v}$ is specified by a *span program* (SP) (M, ρ) along with a tuple of attributes $(v_1, \ldots, v_j)$, and $\boldsymbol{v}(\boldsymbol{x})$ holds iff SP (M, ρ) accepts the truth-value vector of $(\mathsf{T}(x_{i_1} = v_1), \ldots, \mathsf{T}(x_{i_j} = v_j))$. Our scheme can be generalized using non-monotone access structures combined with *inner-product relations* (see Definition 5 and the remark). More precisely, attribute $\boldsymbol{x}$ for signing key $\mathsf{sk}_{\boldsymbol{x}}$ is a tuple of attribute vectors (e.g., $(\overrightarrow{x}_1, \ldots, \overrightarrow{x}_i) \in \mathbb{F}_q^{n_1+\cdots+n_i}$), and predicate $\boldsymbol{v}$ for verification is a non-monotone access structure or span program (SP) (M, ρ) along with a tuple of attribute vectors (e.g., $(\overrightarrow{v}_1, \ldots, \overrightarrow{v}_j) \in \mathbb{F}_q^{n_1+\cdots+n_j}$), where the component-wise inner-product relations for attribute vectors (e.g., $\{\overrightarrow{x}_{i_\iota} \cdot \overrightarrow{v}_\iota = 0$ or not $\}_{\iota \in \{1,\ldots,j\}}$) are input to SP (M, ρ). Namely, $\boldsymbol{v}(\boldsymbol{x})$ holds iff the truth-value vector of $(\mathsf{T}(\overrightarrow{x}_{i_1} \cdot \overrightarrow{v}_1 = 0), \ldots, \mathsf{T}(\overrightarrow{x}_{i_j} \cdot \overrightarrow{v}_j = 0))$ is accepted by SP (M, ρ).
Remark: In our scheme (Section 4), attribute $\boldsymbol{x}$ is expressed by the form $\Gamma := \{(t, x_t) \mid t \in T \subseteq \{1, \ldots, d\}\}$ in place of just an attribute tuple $(x_1, \ldots, x_i)$, where t identifies a sub-universe or category of attributes, and x_t is an attribute in sub-universe t (examples of (t, x_t) are (Name, Alice) and (Age, 38)). Predicate $\boldsymbol{v}$ is expressed by $\mathbb{S} := (M, \rho)$, where ρ is abused as ρ (defined by SP) combined with $\{(t_i, v_i) \mid i = 1, \ldots, \ell\}$ (see Definitions 4 and 5 for the difference regarding ρ in SP and $\mathbb{S}$).
- The proposed ABS scheme is proven to be fully secure under standard assumptions, the *decisional linear (DLIN)* assumption (over prime order pairing groups) and the existence of *collision resistant (CR) hash* functions, in the *standard* model.
- In contrast to the ABS scheme in [19] that employs the Groth-Sahai NIZK protocols, our ABS scheme is more directly constructed without using any general subprotocols like NIZK. Our construction is based on the dual pairing vector spaces (DPVS) proposed by Okamoto and Takashima [20, 21, 14, 22], which can be realized from *any type of (e.g., symmetric or asymmetric)*

prime order bilinear pairing groups. See Section 2.1 for the concept and actual construction of DPVS.

- To prove the security (especially the unforgeability), this paper employs the techniques for fully secure functional encryption (FE) [14, 22], which elaborately combine the dual system encryption methodology proposed by Waters [26] and DPVS.
  Note that although the techniques for the FE schemes in [14, 22] can be employed for ABS, it is still a challenging task to construct a fully secure ABS scheme, since the security requirements of ABS and FE differ in some important points, for example, the privacy condition is required in ABS but there is no counterpart notion in FE. This paper develops several novel techniques for our ABS scheme. See Section 4.1 for more details.
- The efficiency of the proposed ABS scheme is comparable to that of the most efficient ABS scheme in the generic group model [18], and better than that of the only existing fully secure ABS scheme in the standard model [19]. See Section 4.4 for a comparison.
- This paper also presents an extension, multi-authority (MA) setting, of the proposed ABS scheme in Section 5. One of the merits of our MA-ABS scheme is that it is seamlessly extended from the original (single-authority (SA)) setting, in which the signing and verification algorithms of the MA-ABS scheme are essentially the same as those of the original ABS (SA-ABS) scheme.
  In MA-ABS, each authority called an attribute authority is responsible for a category of attributes, and a user obtains a part of secret key for each attribute from an attribute authority responsible for the category of the attribute. We follow the model of MA-ABS introduced in [18, 19], where a central trustee in addition to attribute authorities is required but no interaction among attribute authorities (and the trustee) is necessary, and different attribute authorities may not trust each other, nor even be aware of each other.
  We prove that the proposed MA-ABS scheme is fully secure (in the sense of the MA-ABS model of [18, 19]) under the DLIN assumption and CR hash functions in the standard model (see the full version of this paper for the proof). Our MA-ABS scheme is almost as efficient as the original SA-ABS scheme.

1.3 Related Works

- **Ring and mesh signatures:** Ring and mesh signatures [23, 4] are related to ABS.
  In the ring signatures, the claimed predicate on a signature of message m is that m is endorsed by one of the users identified by the list of public keys $(\mathsf{pk}_1, \mathsf{pk}_2, \ldots)$, or the predicate is a disjunction of a list of public keys. A valid ring signature can be generated by one of the listed users.
  The mesh signatures are an extension of ring signatures, where the predicate is an access structure on a list of pairs comprising a message and public key

(m_i, pk_i), and a valid mesh signature can be generated by a person who has enough standard signatures σ_i on m_i, each valid under pk_i, to satisfy the given access structure.

A crucial difference between mesh signatures and ABS is the security against the collusion of users. In mesh signatures, several users can collude by pooling their signatures together and create signatures that none of them could produce individually. That is, such collusion is considered to be legitimate in mesh signatures. In contrast, the security against collusion attacks is one of the basic requirements in ABS and MA-ABS, as described in Section 1.1 and Section 5.

– **Anonymous credentials (ACs):** Another related concept is ACs [2, 3, 5–8]. The notion of ACs also provides a functionality for users to demonstrate anonymously possession of attributes, but the goals of ACs and ABS differ in several points.

  As mentioned in [19], ACs and ABS aim at different goals: ACs target very strong anonymity even in the registration phase, whereas under less demanding anonymity requirements in the registration phase, ABS aims to achieve more expressive functionalities, more efficient constructions and new applications. In addition, ABS is a signature scheme and a simpler primitive compared with ACs.

1.4 Notations

When A is a random variable or distribution, $y \xleftarrow{\mathsf{R}} A$ denotes that y is randomly selected from A according to its distribution. When A is a set, $y \xleftarrow{\mathsf{U}} A$ denotes that y is uniformly selected from A. $y := z$ denotes that y is set, defined or substituted by z. When a is a fixed value, $A(x) \to a$ (e.g., $A(x) \to 1$) denotes the event that machine (algorithm) A outputs a on input x. A function $f : \mathbb{N} \to \mathbb{R}$ is *negligible* in λ, if for every constant $c > 0$, there exists an integer n such that $f(\lambda) < \lambda^{-c}$ for all $\lambda > n$.

We denote the finite field of order q by $\mathbb{F}_q$, and $\mathbb{F}_q \setminus \{0\}$ by $\mathbb{F}_q^\times$. A vector symbol denotes a vector representation over $\mathbb{F}_q$, e.g., $\overrightarrow{x}$ denotes $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$. For two vectors $\overrightarrow{x} = (x_1, \ldots, x_n)$ and $\overrightarrow{v} = (v_1, \ldots, v_n)$, $\overrightarrow{x} \cdot \overrightarrow{v}$ denotes the inner-product $\sum_{i=1}^n x_i v_i$. The vector $\overrightarrow{0}$ is abused as the zero vector in $\mathbb{F}_q^n$ for any n. X^{T} denotes the transpose of matrix X. A bold face letter denotes an element of vector space $\mathbb{V}$, e.g., $\boldsymbol{x} \in \mathbb{V}$. When $\boldsymbol{b}_i \in \mathbb{V}$ $(i = 1, \ldots, n)$, $\mathsf{span}\langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \rangle \subseteq \mathbb{V}$ (resp. $\mathsf{span}\langle \overrightarrow{x}_1, \ldots, \overrightarrow{x}_n \rangle$) denotes the subspace generated by $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ (resp. $\overrightarrow{x}_1, \ldots, \overrightarrow{x}_n$). For bases $\mathbb{B} := (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N)$ and $\mathbb{B}^* := (\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_N^*)$, $(x_1, \ldots, x_N)_{\mathbb{B}} := \sum_{i=1}^N x_i \boldsymbol{b}_i$ and $(y_1, \ldots, y_N)_{\mathbb{B}^*} := \sum_{i=1}^N y_i \boldsymbol{b}_i^*$.

2 Preliminaries

2.1 Dual Pairing Vector Spaces by Direct Product of Symmetric Pairing Groups

Definition 1. *"Symmetric bilinear pairing groups" $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ are a tuple of a prime q, cyclic additive group $\mathbb{G}$ and multiplicative group $\mathbb{G}_T$ of order q,*

$G \neq 0 \in \mathbb{G}$, and a polynomial-time computable nondegenerate bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ i.e., $e(sG, tG) = e(G, G)^{st}$ and $e(G, G) \neq 1$.

Let $\mathcal{G}_{\mathsf{bpg}}$ be an algorithm that takes input 1^λ and outputs a description of bilinear pairing groups $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ with security parameter λ.

In this paper, we concentrate on the symmetric version of dual pairing vector spaces [20, 21, 14, 22] constructed by using symmetric bilinear pairing groups given in Definition 1.

Definition 2. *"Dual pairing vector spaces (DPVS)" $(q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e)$ by a direct product of symmetric pairing groups $(q, \mathbb{G}, \mathbb{G}_T, G, e)$ are a tuple of prime q, N-dimensional vector space $\mathbb{V} := \overbrace{\mathbb{G} \times \cdots \times \mathbb{G}}^{N}$ over $\mathbb{F}_q$, cyclic group $\mathbb{G}_T$ of order q, canonical basis $\mathbb{A} := (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N)$ of $\mathbb{V}$, where $\boldsymbol{a}_i := (\overbrace{0, \ldots, 0}^{i-1}, G, \overbrace{0, \ldots, 0}^{N-i})$, and pairing $e : \mathbb{V} \times \mathbb{V} \rightarrow \mathbb{G}_T$.*

The pairing is defined by $e(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^{N} e(G_i, H_i) \in \mathbb{G}_T$ where $\boldsymbol{x} := (G_1, \ldots, G_N) \in \mathbb{V}$ and $\boldsymbol{y} := (H_1, \ldots, H_N) \in \mathbb{V}$. This is nondegenerate bilinear i.e., $e(s\boldsymbol{x}, t\boldsymbol{y}) = e(\boldsymbol{x}, \boldsymbol{y})^{st}$ and if $e(\boldsymbol{x}, \boldsymbol{y}) = 1$ for all $\boldsymbol{y} \in \mathbb{V}$, then $\boldsymbol{x} = \boldsymbol{0}$. For all i and j, $e(\boldsymbol{a}_i, \boldsymbol{a}_j) = e(G, G)^{\delta_{i,j}}$ where $\delta_{i,j} = 1$ if $i = j$, and 0 otherwise, and $e(G, G) \neq 1 \in \mathbb{G}_T$.

DPVS also has linear transformations $\phi_{i,j}$ on $\mathbb{V}$ s.t. $\phi_{i,j}(\boldsymbol{a}_j) = \boldsymbol{a}_i$ and $\phi_{i,j}(\boldsymbol{a}_k) = \boldsymbol{0}$ if $k \neq j$, which can be easily achieved by $\phi_{i,j}(\boldsymbol{x}) := (\overbrace{0, \ldots, 0}^{i-1}, G_j, \overbrace{0, \ldots, 0}^{N-i})$ where $\boldsymbol{x} := (G_1, \ldots, G_N)$. We call $\phi_{i,j}$ "canonical maps".

DPVS generation algorithm $\mathcal{G}_{\mathsf{dpvs}}$ takes input 1^λ ($\lambda \in \mathbb{N}$) and $N \in \mathbb{N}$, and outputs a description of $\mathsf{param}_{\mathbb{V}} := (q, \mathbb{V}, \mathbb{G}_T, \mathbb{A}, e)$ with security parameter λ and N-dimensional $\mathbb{V}$. It can be constructed by using $\mathcal{G}_{\mathsf{bpg}}$.

The asymmetric version of DPVS, $(q, \mathbb{V}, \mathbb{V}^*, \mathbb{G}_T, \mathbb{A}, \mathbb{A}^*, e)$, is given in the full version of [22]. The above symmetric version is obtained by identifying $\mathbb{V} = \mathbb{V}^*$ and $\mathbb{A} = \mathbb{A}^*$ in the asymmetric version. (For another construction of DPVS using higher genus Jacobians, see [20].)

2.2 Decisional Linear (DLIN) Assumption

Definition 3 (DLIN Assumption). *The DLIN problem is to guess $\beta \in \{0, 1\}$, given $(\mathsf{param}_{\mathbb{G}},\ G, \xi G, \kappa G, \delta\xi G, \sigma\kappa G, Y_\beta) \xleftarrow{\mathsf{R}} \mathcal{G}_\beta^{\mathsf{DLIN}}(1^\lambda)$, where*

$$\begin{aligned} &\mathcal{G}_\beta^{\mathsf{DLIN}}(1^\lambda) : \mathsf{param}_{\mathbb{G}} := (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda), \\ &\kappa, \delta, \xi, \sigma \xleftarrow{\mathsf{U}} \mathbb{F}_q, \quad Y_0 := (\delta + \sigma)G, \quad Y_1 \xleftarrow{\mathsf{U}} \mathbb{G}, \\ &\text{return} \ \ (\mathsf{param}_{\mathbb{G}},\ G, \xi G, \kappa G, \delta\xi G, \sigma\kappa G, Y_\beta), \end{aligned}$$

for $\beta \xleftarrow{\mathsf{U}} \{0, 1\}$. For a probabilistic machine $\mathcal{E}$, we define the advantage of $\mathcal{E}$ for the DLIN problem as: $\mathsf{Adv}_{\mathcal{E}}^{\mathsf{DLIN}}(\lambda) := \ \Big| \mathsf{Pr}\left[\mathcal{E}(1^\lambda, \varrho) \rightarrow 1 \ \Big| \ \varrho \xleftarrow{\mathsf{R}} \mathcal{G}_0^{\mathsf{DLIN}}(1^\lambda) \right] -$

$\mathsf{Pr}\left[\mathcal{E}(1^{\lambda}, \varrho) \to 1 \,\middle|\, \varrho \xleftarrow{\mathsf{R}} \mathcal{G}_1^{\mathsf{DLIN}}(1^{\lambda})\right]\Big|$. *The DLIN assumption is: For any probabilistic polynomial-time adversary $\mathcal{E}$, the advantage $\mathsf{Adv}_{\mathcal{E}}^{\mathsf{DLIN}}(\lambda)$ is negligible in λ.*

2.3 Collision Resistant (CR) Hash Functions

Let $\lambda \in \mathbb{N}$ be a security parameter. A collision resistant (CR) hash function family, H, associated with $\mathcal{G}_{\mathsf{bpg}}$ and a polynomial, $poly(\cdot)$, specifies two items:

- A family of key spaces indexed by λ. Each such key space is a probability space on bit strings denoted by KH_{λ}. There must exist a probabilistic polynomial-time algorithm whose output distribution on input 1^{λ} is equal to KH_{λ}.
- A family of hash functions indexed by λ, $\mathsf{hk} \xleftarrow{\mathsf{R}} \mathsf{KH}_{\lambda}$ and $\mathsf{D} := \{0,1\}^{poly(\lambda)}$. Each such hash function $\mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}$ maps an element of D to an element of $\mathbb{F}_q^{\times}$ with q that is the first element of output $\mathsf{param}_{\mathbb{G}}$ of $\mathcal{G}_{\mathsf{bpg}}(1^{\lambda})$. There must exist a deterministic polynomial-time algorithm that on input 1^{λ}, hk and $\varrho \in \mathsf{D}$, outputs $\mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}(\varrho)$.

Let $\mathcal{E}$ be a probabilistic polynomial-time machine. For all λ, we define $\mathsf{Adv}_{\mathcal{E}}^{\mathsf{H},\mathsf{CR}}(\lambda) :=\ \Pr[(\varrho_1, \varrho_2) \in \mathsf{D}^2 \wedge \varrho_1 \neq \varrho_2 \wedge \mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}(\varrho_1) = \mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}(\varrho_2)]$, where $\mathsf{D} := \{0,1\}^{poly(\lambda)}, \mathsf{hk} \xleftarrow{\mathsf{R}} \mathsf{KH}_{\lambda}$, and $(\varrho_1, \varrho_2) \xleftarrow{\mathsf{R}} \mathcal{E}(1^{\lambda}, \mathsf{hk}, \mathsf{D})$. H is a collision resistant (CR) hash function family if for any probabilistic polynomial-time adversary $\mathcal{E}$, $\mathsf{Adv}_{\mathcal{E}}^{\mathsf{H},\mathsf{CR}}(\lambda)$ is negligible in λ.

3 ABS for Non-Monotone Predicates

3.1 Span Programs and Non-Monotone Access Structures

Definition 4 (Span Programs [1]). *Let $\{p_1, \ldots, p_n\}$ be a set of variables. A span program over $\mathbb{F}_q$ is a labeled matrix, $\hat{M} := (M, \rho)$, where M is a $(\ell \times r)$ matrix over $\mathbb{F}_q$ and ρ is a labeling of the rows of M by literals from $\{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$ (every row is labeled by one literal), i.e., $\rho : \{1, \ldots, \ell\} \to \{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$.*

A span program accepts or rejects an input by the following criterion. For every input sequence $\delta \in \{0,1\}^n$ define submatrix M_δ of M consisting of those rows whose labels are set to 1 by the input δ, i.e., either rows labeled by some p_i such that $\delta_i = 1$ or rows labeled by some by some $\neg p_i$ such that $\delta_i = 0$. (i.e., $\gamma : \{1, \ldots, \ell\} \to \{0,1\}$ is defined by $\gamma(j) = 1$ if $[\rho(j) = p_i] \wedge [\delta_i = 1]$ or $[\rho(j) = \neg p_i] \wedge [\delta_i = 0]$, and $\gamma(j) = 0$ otherwise. $M_\delta := (M_j)_{\gamma(j)=1}$, where M_j is the j-th row of M.)

Span program $\hat{M}$ accepts δ if and only if $\overrightarrow{1} \in \mathsf{span}\langle M_\delta \rangle$, i.e., some linear combination of the rows of M_δ gives the all one vector, $\overrightarrow{1}$. (The row vector has the value 1 in each coordinate.) A span program computes boolean function f if it accepts exactly those inputs δ where $f(\delta) = 1$.

A span program is called monotone if the labels of the rows are only the positive literals $\{p_1, \ldots, p_n\}$. Monotone span programs compute monotone functions. (So, a span program in general is "non"-monotone.)

We assume that access structure matrix M (of type $\ell \times r$) satisfies the condition: $M_i \neq \overrightarrow{0}$ for $i = 1, \ldots, \ell$.

We now introduce a non-monotone access structure with evaluating map γ by using the inner-product of attribute vectors in a general form. Although we will show the notion, security definition and security proof of the proposed ABS scheme in this general form, we will describe the proposed ABS scheme in a simpler form in Section 4.2. We will show this simpler form of Definition 5 in the remark.

Definition 5 (Inner-Products of Attribute Vectors and Access Structures). *$\mathcal{U}_t$ ($t = 1, \ldots, d$ and $\mathcal{U}_t \subset \{0,1\}^*$) is a sub-universe, a set of attributes, each of which is expressed by a pair of sub-universe id and n_t-dimensional vector, i.e., $(t, \overrightarrow{v})$, where $t \in \{1, \ldots, d\}$ and $\overrightarrow{v} \in \mathbb{F}_q^{n_t} \setminus \{\overrightarrow{0}\}$.*

We now define such an attribute to be a variable, p, of span program $\hat{M} := (M, \rho)$ i.e., $p := (t, \overrightarrow{v})$. Access structure $\mathbb{S}$ is span program $\hat{M} := (M, \rho)$ along with variables $p := (t, \overrightarrow{v}), p' := (t', \overrightarrow{v}'), \ldots$, i.e., $\mathbb{S} := (M, \rho)$ such that $\rho : \{1, \ldots, \ell\} \to \{(t, \overrightarrow{v}), (t', \overrightarrow{v}'), \ldots, \neg(t, \overrightarrow{v}), \neg(t', \overrightarrow{v}'), \ldots\}$.

Let Γ be a set of attributes, i.e., $\Gamma := \{(t, \overrightarrow{x}_t) \mid \overrightarrow{x}_t \in \mathbb{F}_q^{n_t} \setminus \{\overrightarrow{0}\}, 1 \le t \le d\}$.

When Γ is given to access structure $\mathbb{S}$, map $\gamma : \{1, \ldots, \ell\} \to \{0,1\}$ for span program $\hat{M} := (M, \rho)$ is defined as follows: For $i = 1, \ldots, \ell$, set $\gamma(i) = 1$ if $[\rho(i) = (t, \overrightarrow{v}_i)] \wedge [(t, \overrightarrow{x}_t) \in \Gamma] \wedge [\overrightarrow{v}_i \cdot \overrightarrow{x}_t = 0]$ or $[\rho(i) = \neg(t, \overrightarrow{v}_i)] \wedge [(t, \overrightarrow{x}_t) \in \Gamma] \wedge [\overrightarrow{v}_i \cdot \overrightarrow{x}_t \neq 0]$. Set $\gamma(i) = 0$ otherwise.

Access structure $\mathbb{S} := (M, \rho)$ accepts Γ iff $\overrightarrow{1} \in \mathsf{span}\langle (M_i)_{\gamma(i)=1} \rangle$.

Remark: A simpler form of the inner-product relations in the above-mentioned access structures is a special case when $n_t = 2$ for all $t \in \{1, \ldots, d\}$, and $\overrightarrow{x} := (1, x)$ and $\overrightarrow{v} := (v, -1)$. Hence, $(t, \overrightarrow{x}_t) := (t, (1, x_t))$ and $(t, \overrightarrow{v}_i) := (t, (v_i, -1))$, but we often denote them shortly by (t, x_t) and (t, v_i). Then, $\mathbb{S} := (M, \rho)$ such that $\rho : \{1, \ldots, \ell\} \to \{(t, v), (t', v'), \ldots \neg(t, v), \neg(t', v'), \ldots\}$ $(v, v', \ldots \in \mathbb{F}_q)$, and $\Gamma := \{(t, x_t) \mid x_t \in \mathbb{F}_q, 1 \le t \le d\}$.

When Γ is given to access structure $\mathbb{S}$, map $\gamma : \{1, \ldots, \ell\} \to \{0,1\}$ for span program $\hat{M} := (M, \rho)$ is defined as follows: For $i = 1, \ldots, \ell$, set $\gamma(i) = 1$ if $[\rho(i) = (t, v_i)] \wedge [(t, x_t) \in \Gamma] \wedge [v_i = x_t]$ or $[\rho(i) = \neg(t, v_i)] \wedge [(t, x_t) \in \Gamma] \wedge [v_i \neq x_t]$. Set $\gamma(i) = 0$ otherwise.

We now construct a secret-sharing scheme for a (non-monotone) access structure (span program).

Definition 6. *A secret-sharing scheme for access structure $\mathbb{S} := (M, \rho)$ is:*

1. *Let M be an $\ell \times r$ matrix, and column vector $\overrightarrow{f}^{\mathrm{T}} := (f_1, \ldots, f_r)^{\mathrm{T}} \xleftarrow{\mathsf{U}} \mathbb{F}_q^r$. Then, $s_0 := \overrightarrow{1} \cdot \overrightarrow{f}^{\mathrm{T}} = \sum_{k=1}^{r} f_k$ is the secret to be shared, and $\overrightarrow{s}^{\mathrm{T}} := (s_1, \ldots, s_\ell)^{\mathrm{T}} := M \cdot \overrightarrow{f}^{\mathrm{T}}$ is the vector of ℓ shares of secret s_0 and share s_i belongs to $\rho(i)$.*

2. If access structure $\mathbb{S} := (M, \rho)$ accepts Γ, i.e., $\overrightarrow{1} \in \mathsf{span}\langle (M_i)_{\gamma(i)=1} \rangle$ with $\gamma : \{1, \ldots, \ell\} \to \{0, 1\}$, then there exist constants $\{\alpha_i \in \mathbb{F}_q \mid i \in I\}$ such that $I \subseteq \{i \in \{1, \ldots, \ell\} \mid \gamma(i) = 1\}$ and $\sum_{i \in I} \alpha_i s_i = s_0$. Furthermore, these constants $\{\alpha_i\}$ can be computed in time polynomial in the size of matrix M.

3.2 Definitions and Security of ABS

Definition 7 (Attribute-Based Signatures : ABS). *An attribute-based signature scheme consists of four algorithms.*

Setup *This is a randomized algorithm that takes as input security parameter and format $\overrightarrow{n} := (d; n_1, \ldots, n_d)$ of attributes. It outputs public parameters* pk *and master key* sk.

KeyGen *This is a randomized algorithm that takes as input a set of attributes, $\Gamma := \{(t, \overrightarrow{x}_t) | \overrightarrow{x}_t \in \mathbb{F}_q^{n_t} \setminus \{\overrightarrow{0}\}, 1 \leq t \leq d\}$,* pk *and* sk*. It outputs signature generation key* sk_Γ.

Sig *This is a randomized algorithm that takes as input message m, access structure $\mathbb{S} := (M, \rho)$, signature generation key sk_Γ, and public parameters* pk *such that $\mathbb{S}$ accepts Γ. It outputs signature σ.*

Ver *This takes as input message m, access structure $\mathbb{S}$, signature σ and public parameters* pk*. It outputs boolean value* accept $:= 1$ *or* reject $:= 0$.

An ABS scheme should have the following correctness property: for all $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\mathsf{R}} \mathsf{Setup}(1^\lambda, \overrightarrow{n})$, all messages m, all attribute sets Γ, all signing keys $\mathsf{sk}_\Gamma \xleftarrow{\mathsf{R}} \mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \Gamma)$, all access structures $\mathbb{S}$ such that $\mathbb{S}$ accepts Γ, and all signatures $\sigma \xleftarrow{\mathsf{R}} \mathsf{Sig}(\mathsf{pk}, \mathsf{sk}_\Gamma, m, \mathbb{S})$, it holds that $\mathsf{Ver}(\mathsf{pk}, m, \mathbb{S}, \sigma) = 1$ with probability 1.

Definition 8 (Perfect Privacy). *An ABS scheme is perfectly private, if, for all $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\mathsf{R}} \mathsf{Setup}(1^\lambda, \overrightarrow{n})$, all messages m, all attribute sets Γ_1 and Γ_2, all signing keys $\mathsf{sk}_{\Gamma_1} \xleftarrow{\mathsf{R}} \mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \Gamma_1)$ and $\mathsf{sk}_{\Gamma_2} \xleftarrow{\mathsf{R}} \mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \Gamma_2)$, all access structures $\mathbb{S}$ such that $\mathbb{S}$ accepts Γ_1 and $\mathbb{S}$ accepts Γ_2, distributions $\mathsf{Sig}(\mathsf{pk}, \mathsf{sk}_{\Gamma_1}, m, \mathbb{S})$ and $\mathsf{Sig}(\mathsf{pk}, \mathsf{sk}_{\Gamma_2}, m, \mathbb{S})$ are equal.*

For an ABS scheme with prefect privacy, we define algorithm $\mathsf{AltSig}(\mathsf{pk}, \mathsf{sk}, m, \mathbb{S})$ with $\mathbb{S}$ and master key sk instead of Γ and sk_Γ: First, generate $\mathsf{sk}_\Gamma \xleftarrow{\mathsf{R}} \mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \Gamma)$ for arbitrary Γ which satisfies $\mathbb{S}$, then $\sigma \xleftarrow{\mathsf{R}} \mathsf{Sig}(\mathsf{pk}, \mathsf{sk}_\Gamma, m, \mathbb{S})$. return σ.

Since the correct distribution on signatures can be perfectly simulated without taking any private information as input, signatures must not leak any such private information of the signer.

Definition 9 (Unforgeability). *For an adversary, $\mathcal{A}$, we define $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{ABS,UF}}(\lambda)$ to be the success probability in the following experiment for any security parameter λ. An ABS scheme is existentially unforgeable if the success probability of any polynomial-time adversary is negligible:*

1. *Run* $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\mathsf{R}} \mathsf{Setup}(1^{\lambda}, \vec{n})$ *and give* pk *to the adversary.*
2. *The adversary is given access to oracles* $\mathsf{KeyGen}(\mathsf{pk}, \mathsf{sk}, \cdot)$ *and* $\mathsf{AltSig}(\mathsf{pk}, \mathsf{sk}, \cdot, \cdot)$.
3. *At the end, the adversary outputs* $(m', \mathbb{S}', \sigma')$.

We say the adversary succeeds if $(m', \mathbb{S}')$ *was never queried to the* AltSig *oracle,* $\mathbb{S}'$ *does not accept any* Γ *queried to the* KeyGen *oracle, and* $\mathsf{Ver}(\mathsf{pk}, m', \mathbb{S}', \sigma') = 1$.

4 Proposed ABS Scheme

4.1 Construction Ideas

Here, we will show some basic ideas to construct the proposed ABS scheme. Our ABS scheme is constructed on a ciphertext policy (CP) functional encryption (FE) scheme [22], which is adaptively payload-hiding against chosen-plaintext attacks. The description of the CP-FE scheme is given in the full version of [22].

Roughly speaking, a secret signing key, sk_{Γ}, with attribute set Γ and a verification text, $\vec{\boldsymbol{c}}$, with access structure $\mathbb{S}$ (for signature verification) in our ABS scheme correspond to a secret decryption key, sk_{Γ}, with Γ and a ciphertext, $\vec{\boldsymbol{c}}$, with $\mathbb{S}$ in the CP-FE scheme, respectively. No counterpart of a signature, $\vec{\boldsymbol{s}}^*$, in the ABS exists in the CP-FE, and the privacy property for signature $\vec{\boldsymbol{s}}^*$ is also specific in ABS. Signature $\vec{\boldsymbol{s}}^*$ in ABS may be interpreted to be a decryption key specialized to decrypt a ciphertext with access structure $\mathbb{S}$, that is delegated from secret key sk_{Γ}.

The algorithms of the proposed ABS scheme can be described in the light of such correspondence to the CP-FE scheme:

Setup Almost the same as that in the CP-FE scheme except that $\{\widehat{\mathbb{B}}^*_t\}_{t=1,\ldots,d+1}$ are revealed as a *public* parameter in our ABS, while they are *secret* in the CP-FE scheme. They are published in our ABS for the signature generation procedure Sig to meet the *privacy* of signers (for randomization). This implies an important gap between CP-FE and ABS.

KeyGen Almost the same as that in the CP-FE scheme except that a (7 dimensional) space with basis $\mathbb{B}^*_{d+1}$ is additionally introduced in our ABS and two elements $\boldsymbol{k}^*_{d+1,1}$ and $\boldsymbol{k}^*_{d+1,2}$ in this space are included in a secret signing key in order to embed the hash value, $\mathsf{H}^{\lambda,\mathsf{D}}_{\mathsf{hk}}(m \,||\, \mathbb{S})$, of message m and access structure $\mathbb{S}$ in signature $\vec{\boldsymbol{s}}^*$.

Sig Specific in ABS. To meet the privacy condition for $\vec{\boldsymbol{s}}^*$, a novel technique is employed to randomly generate a signature from sk_{Γ} and $\{\widehat{\mathbb{B}}^*_t\}_{t=1,\ldots,d+1}$.

Ver Signature $\vec{\boldsymbol{s}}^*$ in the ABS is an endorsement to message m by a signer with attributes accepted by access structure $\mathbb{S}$. The signature verification in our ABS checks whether signature (or specific decryption key) $\vec{\boldsymbol{s}}^*$ works as a decryption key to decrypt a verification text (or a ciphertext) associated with $\mathbb{S}$ and $\mathsf{H}^{\lambda,\mathsf{D}}_{\mathsf{hk}}(m \,||\, \mathbb{S})$.

Security proofs Roughly speaking, the *adaptive*-predicate unforgeability of the ABS under the KeyGen oracle attacks can be guaranteed by the *adaptive* payload-hiding property of the CP-FE, since a forged signature implies a decryption key specified for the challenge ciphertext to break the payload-hiding. Note that there are many subtleties in the proof of unforgeability for the ABS, e.g., the unforgeability should be ensured in the ABS even when publishing $\{\widehat{\mathbb{B}}^*_t\}_{t=1,\ldots,d+1}$ for the privacy requirement, while they are secret in the CP-FE. We develop a novel technique to resolve the difficulty. See the full version of this paper for more details.

4.2 Construction

For simplicity, here, we describe our ABS scheme for a specific parameter $\vec{n} := (d; 2, \ldots, 2)$ (see the remark of Definition 5). A general form of our ABS scheme is given in the full version.

We define function $\widetilde{\rho} : \{1,\ldots,\ell\} \to \{1,\ldots,d\}$ by $\widetilde{\rho}(i) := t$ if $\rho(i) = (t,v)$ or $\rho(i) = \neg(t,v)$, where ρ is given in access structure $\mathbb{S} := (M,\rho)$. In the proposed scheme, we assume that $\widetilde{\rho}$ is injective for $\mathbb{S} := (M,\rho)$. The full version of this paper shows how to relax the restriction.

```
$\mathsf{Setup}(1^\lambda,\ \vec{n} := (d;2,\ldots,2))$ : $\mathsf{param}_{\mathbb{G}} := (q,\mathbb{G},\mathbb{G}_T,G,e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda)$,
  $\mathsf{hk} \xleftarrow{\mathsf{R}} \mathsf{KH}_\lambda,\ \psi \xleftarrow{\mathsf{U}} \mathbb{F}_q^\times,\ N_0 := 4,\ N_t := 7$ for $t = 1,\ldots,d+1$,
  for $t = 0,\ldots,d+1,\ \ \mathsf{param}_{\mathbb{V}_t} := (q,\mathbb{V}_t,\mathbb{G}_T,\mathbb{A}_t,e) := \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, N_t, \mathsf{param}_{\mathbb{G}})$,
    $X_t := (\chi_{t,i,j})_{i,j} \xleftarrow{\mathsf{U}} GL(N_t,\mathbb{F}_q),\ (\vartheta_{t,i,j})_{i,j} := \psi \cdot (X_t^{-1})^{\mathrm{T}}$,
    $\boldsymbol{b}_{t,i} := (\chi_{t,i,1},\ldots,\chi_{t,i,N_t})_{\mathbb{A}_t},\ \mathbb{B}_t := (\boldsymbol{b}_{t,1},\ldots,\boldsymbol{b}_{t,N_t})$,
    $\boldsymbol{b}^*_{t,i} := (\vartheta_{t,i,1},\ldots,\vartheta_{t,i,N_t})_{\mathbb{A}_t},\ \mathbb{B}^*_t := (\boldsymbol{b}^*_{t,1},\ldots,\boldsymbol{b}^*_{t,N_t})$,
  $g_T := e(G,G)^\psi,\ \ \mathsf{param}_{\vec{n}} := (\{\mathsf{param}_{\mathbb{V}_t}\}_{t=0,\ldots,d+1},\ g_T)$,
  $\widehat{\mathbb{B}}_0 := (\boldsymbol{b}_{0,1},\boldsymbol{b}_{0,4}),\ \ \widehat{\mathbb{B}}_t := (\boldsymbol{b}_{t,1},\boldsymbol{b}_{t,2},\boldsymbol{b}_{t,7})$ for $t = 1,\ldots,d+1$,
  $\widehat{\mathbb{B}}^*_t := (\boldsymbol{b}^*_{t,1},\boldsymbol{b}^*_{t,2},\boldsymbol{b}^*_{t,5},\boldsymbol{b}^*_{t,6})$ for $t = 1,\ldots,d+1$,
  $\mathsf{sk} := \boldsymbol{b}^*_{0,1},\ \ \ \mathsf{pk} := (1^\lambda, \mathsf{hk}, \mathsf{param}_{\vec{n}}, \{\widehat{\mathbb{B}}_t\}_{t=0,\ldots,d+1}, \{\widehat{\mathbb{B}}^*_t\}_{t=1,\ldots,d+1}, \boldsymbol{b}^*_{0,3})$.
  return $\mathsf{sk},\ \mathsf{pk}$.
$\mathsf{KeyGen}(\mathsf{pk},\ \mathsf{sk},\ \ \Gamma := \{(t,x_t) \mid 1 \le t \le d\})$ :
  $\delta \xleftarrow{\mathsf{U}} \mathbb{F}_q^\times,\ \ \varphi_0, \varphi_{t,\iota}, \varphi_{d+1,1,\iota}, \varphi_{d+1,2,\iota} \xleftarrow{\mathsf{U}} \mathbb{F}_q$ for $t = 1,\ldots,d;\ \iota = 1,2$;
  $\boldsymbol{k}^*_0 := (\ \delta,\ 0,\ \varphi_0,\ 0\ )_{\mathbb{B}^*_0}$,
  $\boldsymbol{k}^*_t := (\delta(\ 1,\ x_t\ ),\ \ 0,\ 0,\ \ \varphi_{t,1},\ \varphi_{t,2},\ \ 0)_{\mathbb{B}^*_t}$ for $(t,x_t) \in \Gamma$,
  $\boldsymbol{k}^*_{d+1,1} := (\ \delta(\ 1,\ 0\ ),\ \ 0,\ 0,\ \ \varphi_{d+1,1,1},\ \varphi_{d+1,1,2},\ \ 0\ )_{\mathbb{B}^*_{d+1}}$,
  $\boldsymbol{k}^*_{d+1,2} := (\ \delta(\ 0,\ 1\ ),\ \ 0,\ 0,\ \ \varphi_{d+1,2,1},\ \varphi_{d+1,2,2},\ \ 0\ )_{\mathbb{B}^*_{d+1}}$,
  $T := \{0,(d+1,1),(d+1,2)\} \cup \{t\ \mid\ 1 \le t \le d,\ (t,x_t) \in \Gamma\}$,
  return $\mathsf{sk}_\Gamma := (\Gamma, \{\boldsymbol{k}^*_t\}_{t \in T})$.
$\mathsf{Sig}(\mathsf{pk},\ \mathsf{sk}_\Gamma,\ m,\ \mathbb{S} := (M,\rho))$ : If $\mathbb{S} := (M,\rho)$ accepts $\Gamma := \{(t,x_t)\}$, then
  compute $I$ and $\{\alpha_i\}_{i \in I}$ such that $\sum_{i \in I} \alpha_i M_i = \vec{1}$, and
  $I \subseteq \{i \in \{1,\ldots,\ell\} \mid\ [\rho(i) = (t,v_i)\ \wedge\ (t,x_t) \in \Gamma\ \wedge\ v_i = x_t]$
```

$$\vee\ \ [\rho(i) = \neg(t, v_i)\ \wedge\ (t, x_t) \in \Gamma\ \wedge\ v_i \neq x_t]\ \},$$

$\xi \xleftarrow{\mathsf{U}} \mathbb{F}_q^{\times},\ \ (\beta_i) \xleftarrow{\mathsf{U}} \{(\beta_1, \ldots, \beta_\ell)\ \mid\ \sum_{i=1}^{\ell} \beta_i M_i = \vec{0}\},$

$\boldsymbol{s}_0^* := \xi \boldsymbol{k}_0^* + \boldsymbol{r}_0^*,$ where $\boldsymbol{r}_0^* \xleftarrow{\mathsf{U}} \mathsf{span}\langle \boldsymbol{b}_{0,3}^* \rangle,$

$\boldsymbol{s}_i^* := \gamma_i \cdot \xi \boldsymbol{k}_t^* + \sum_{\iota=1}^{2} y_{i,\iota} \cdot \boldsymbol{b}_{t,\iota}^* + \boldsymbol{r}_i^* \quad$ for $1 \leq i \leq \ell,$

where $\boldsymbol{r}_i^* \xleftarrow{\mathsf{U}} \mathsf{span}\langle \boldsymbol{b}_{t,5}^*, \boldsymbol{b}_{t,6}^* \rangle,$ and $\gamma_i, \vec{y}_i := (y_{i,1}, y_{i,2})$ are defined as

if $i \in I\ \wedge\ \rho(i) = (t, v_i), \quad \gamma_i := \alpha_i, \ \ \vec{y}_i := \beta_i(1, v_i),$

if $i \in I\ \wedge\ \rho(i) = \neg(t, v_i), \quad \gamma_i := \dfrac{\alpha_i}{v_i - x_t}, \ \ \vec{y}_i := \dfrac{\beta_i}{v_i - y_i}(1, y_i),$

where $y_i \xleftarrow{\mathsf{U}} \mathbb{F}_q \backslash \{v_i\},$

if $i \notin I\ \wedge\ \rho(i) = (t, v_i), \quad \gamma_i := 0, \ \ \vec{y}_i := \beta_i(1, v_i),$

if $i \notin I\ \wedge\ \rho(i) = \neg(t, v_i), \quad \gamma_i := 0, \ \ \vec{y}_i := \dfrac{\beta_i}{v_i - y_i}(1, y_i),$

where $y_i \xleftarrow{\mathsf{U}} \mathbb{F}_q \backslash \{v_i\},$

$\boldsymbol{s}_{\ell+1}^* := \xi(\boldsymbol{k}_{d+1,1}^* + \mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}(m \,||\, \mathbb{S}) \cdot \boldsymbol{k}_{d+1,2}^*) + \boldsymbol{r}_{\ell+1}^*,$

where $\boldsymbol{r}_{\ell+1}^* \xleftarrow{\mathsf{U}} \mathsf{span}\langle \boldsymbol{b}_{d+1,5}^*, \boldsymbol{b}_{d+1,6}^* \rangle,$

return $\vec{\boldsymbol{s}}^* := (\boldsymbol{s}_0^*, \ldots, \boldsymbol{s}_{\ell+1}^*).$

$\mathsf{Ver}(\mathsf{pk},\ m,\ \mathbb{S} := (M, \rho), \vec{\boldsymbol{s}}^*):\ \ \vec{f} \xleftarrow{\mathsf{U}} \mathbb{F}_q^{r},\ \vec{s}^{\mathrm{T}} := (s_1, \ldots, s_\ell)^{\mathrm{T}} := M \cdot \vec{f}^{\mathrm{T}},$

$s_0 := \vec{1} \cdot \vec{f}^{\mathrm{T}},\ \eta_0, \eta_{\ell+1}, \theta_{\ell+1}, s_{\ell+1} \xleftarrow{\mathsf{U}} \mathbb{F}_q,$

$\boldsymbol{c}_0 := (\ -s_0 - s_{\ell+1},\ 0,\ 0,\ \eta_0\ )_{\mathbb{B}_0},$

for $1 \leq i \leq \ell,$

if $\rho(i) = (t, v_i),$ return 0 if $\boldsymbol{s}_i^* \notin \mathbb{V}_t,$ else

$\boldsymbol{c}_i := (\ s_i + \theta_i v_i,\ -\theta_i,\ \ 0,\ 0,\ \ 0,\ 0,\ \ \eta_i\ )_{\mathbb{B}_t},$ where $\theta_i, \eta_i \xleftarrow{\mathsf{U}} \mathbb{F}_q,$

if $\rho(i) = \neg(t, v_i),$ return 0 if $\boldsymbol{s}_i^* \notin \mathbb{V}_t,$ else

$\boldsymbol{c}_i := (\ s_i(\ v_i,\ -1\ ),\ \ 0,\ 0,\ \ 0,\ 0,\ \ \eta_i\ )_{\mathbb{B}_t},$ where $\eta_i \xleftarrow{\mathsf{U}} \mathbb{F}_q,$

$\boldsymbol{c}_{\ell+1} := (\ s_{\ell+1} - \theta_{\ell+1} \cdot \mathsf{H}_{\mathsf{hk}}^{\lambda,\mathsf{D}}(m \,||\, \mathbb{S}),\ \theta_{\ell+1},\ \ 0,\ 0,\ \ 0,\ 0,\ \ \eta_{\ell+1}\ )_{\mathbb{B}_{d+1}},$

return 0 if $e(\boldsymbol{b}_{0,1}, \boldsymbol{s}_0^*) = 1,$

return 1 if $\prod_{i=0}^{\ell+1} e(\boldsymbol{c}_i, \boldsymbol{s}_i^*) = 1,$ return 0 otherwise.

[Correctness] $\prod_{i=0}^{\ell+1} e(\boldsymbol{c}_i, \boldsymbol{s}_i^*)$
$= e(\boldsymbol{c}_0, \boldsymbol{k}_0^*)^{\xi} \cdot \prod_{i \in I} e(\boldsymbol{c}_i, \boldsymbol{k}_t^*)^{\gamma_i \xi} \cdot \prod_{i=1}^{\ell} \prod_{\iota=1}^{2} e(\boldsymbol{c}_i, \boldsymbol{b}_{t,\iota}^*)^{y_{i,\iota}} \cdot e(\boldsymbol{c}_{\ell+1}, \boldsymbol{s}_{\ell+1}^*)$
$= g_T^{\xi\delta(-s_0 - s_{\ell+1})} \cdot \prod_{i \in I} g_T^{\xi\delta\alpha_i s_i} \cdot \prod_{i=1}^{\ell} g_T^{\beta_i s_i} \cdot g_T^{\xi\delta s_{\ell+1}}$
$= g_T^{\xi\delta(-s_0 - s_{\ell+1})} \cdot g_T^{\xi\delta s_0} \cdot g_T^{\xi\delta s_{\ell+1}} = 1.$

4.3 Security

Theorem 1. *The proposed ABS scheme is perfectly private.*

Theorem 2. *The proposed ABS scheme is unforgeable (adaptive-predicate unforgeable) under the DLIN assumption and the existence of collision resistant hash functions.*

For any adversary $\mathcal{A}$, there exist probabilistic machines $\mathcal{E}_1, \mathcal{E}^+_{2,h}$, $\mathcal{E}_{2,h+1}$ $(h = 0, \ldots, \nu_1 - 1)$, $\mathcal{E}_{3,h}, \mathcal{E}_{4,h}$ $(h = 1, \ldots, \nu_2)$, whose running times are essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ABS,UF}}_{\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{E}_1}(\lambda) + \textstyle\sum_{h=0}^{\nu_1-1} \left(\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{E}^+_{2,h}}(\lambda) + \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{E}_{2,h+1}}(\lambda)\right) \\ + \textstyle\sum_{h=1}^{\nu_2} \left(\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{E}_{3,h}}(\lambda) + \mathsf{Adv}^{\mathsf{H,CR}}_{\mathcal{E}_{4,h}}(\lambda)\right) + \epsilon,\end{aligned}$$

where ν_1 is the maximum number of $\mathcal{A}$'s KeyGen *queries, ν_2 is the maximum number of $\mathcal{A}$'s* AltSig *queries, and $\epsilon := ((2d + 16)\nu_1 + 18\nu_2 + 2d + 18)/q$.*

The proofs of Theorems 1 and 2 are given in the full version of this paper.

4.4 Performance

In this section, we compare the efficiency and security of the proposed ABS scheme with the existing ABS schemes in the standard model (two typical instantiations) [19] as well as the ABS scheme in the generic group model [18] (as a benchmark). Since all of these schemes can be implemented over a *prime order* pairing group, the size of a group element can be around the size of $\mathbb{F}_q$ (e.g., 256 bits). In Table 1, ℓ and r represent the size of the underlying access structure matrix M for a predicate, i.e., $M \in \mathbb{F}_q^{\ell \times r}$. For example, some predicate with 4

Table 1. Comparison with the Existing ABS Schemes

	MPR08 [18]	MPR10 [19] (Boneh-Boyen based)	MPR10 [19] (Waters based)	Proposed
Signature size (# of group elts)	$\ell + r + 2$	$51\ell + 2r + 18\lambda\ell$	$36\ell + 2r$ $+9\lambda + 12$	$7\ell + 11$
Model	generic group model	standard model	standard model	standard model
Security	full	full	full	full
Assumptions	CR hash	q-SDH and DLIN	DLIN	DLIN and CR hash
Predicates	monotone	monotone	monotone	non-monotone
Sig. size example 1 ($\ell = 10, r = 5,$ $\lambda = 128$)	17	23560	1534	81
Sig. size example 2 ($\ell = 100, r = 50,$ $\lambda = 128$)	152	282400	4864	711

AND and 5 OR gates as well as 10 variables may be expressed by a 10×5 matrix, and a predicate with 49 AND and 50 OR gates as well as 100 variables may be expressed by a 100×50 matrix (see the appendix of [15]). λ is the security parameter (e.g., 128).

5 Multi-Authority ABS (MA-ABS)

5.1 Definitions and Security of MA-ABS

We follow the model and security definitions of MA-ABS in [18, 19].

Definition 10 (Multi-Authority ABS : MA-ABS). *A multi-authority ABS scheme consists of the following algorithms/protocols.*

TSetup *This is a randomized algorithm. The signature trustee runs algorithm* $\mathsf{TSetup}(1^\lambda)$ *which outputs trustee public key* tpk *and trustee secret key* tsk*.*

$\mathsf{UserReg}$ *This is a randomized algorithm. When a user with user id* uid *registers with the signature trustee, the trustee runs* $\mathsf{UserReg}(\mathsf{tpk}, \mathsf{tsk}, \mathsf{uid})$ *which outputs public user-token* $\mathsf{token}_{\mathsf{uid}}$*. The trustee gives* $\mathsf{token}_{\mathsf{uid}}$ *to the user.*

ASetup *This is a randomized algorithm. Attribute authority t ($1 \leq t \leq d$) who wishes to issue attributes runs* $\mathsf{ASetup}(\mathsf{tpk})$ *which outputs attribute-authority public key* apk_t *and attribute-authority secret key* ask_t*. The attribute authority, t, publishes* apk_t *and stores* ask_t*.*

$\mathsf{AttrGen}$ *This is a randomized algorithm. When attribute authority t issues user* uid *a secret key associated with attribute x_t, first it obtains (from the user) her user-token* $\mathsf{token}_{\mathsf{uid}}$*, and runs token verification algorithm* $\mathsf{TokenVerify}(\mathsf{tpk}, \mathsf{uid}, \mathsf{token}_{\mathsf{uid}})$*. If the token is verified, then it runs* $\mathsf{AttrGen}(\mathsf{tpk}, t, \mathsf{ask}_t, \mathsf{token}_{\mathsf{uid}}, x_t)$ *that outputs attribute secret key* usk_t*. The attribute authority gives* usk_t *to the user.*

Sig *This is a randomized algorithm. A user signs message m with claim-predicate (access structure) $\mathbb{S} := (M, \rho)$, only if there is a set of attributes Γ such that $\mathbb{S}$ accepts Γ, the user has obtained a set of keys* $\{\mathsf{usk}_t \mid (t, x_t) \in \Gamma\}$ *from the attribute authorities. Then signature σ can be generated using* $\mathsf{Sig}(\mathsf{tpk}, \mathsf{token}_{\mathsf{uid}}, \{\mathsf{apk}_t, \mathsf{usk}_t \mid (t, x_t) \in \Gamma\}, m, \mathbb{S})$*, where* $\mathsf{usk}_t \overset{\mathsf{R}}{\leftarrow} \mathsf{AttrGen}(\mathsf{tpk}, t, \mathsf{ask}_t, \mathsf{token}_{\mathsf{uid}}, x_t)$*.*

Ver *To verify signature σ on message m with claim-predicate (access structure) $\mathbb{S}$, a user runs* $\mathsf{Ver}(\mathsf{tpk}, \{\mathsf{apk}_t\}, m, \mathbb{S}, \sigma)$ *which outputs boolean value* $\mathsf{accept} := 1$ *or* $\mathsf{reject} := 0$*.*

The definition of perfect privacy for the multi-authority (MA) ABS is essentially the same as that of the single-authority (SA) ABS (Definition 8). The major difference of the unforgeability of MA-ABS and SA-ABS is that adversary $\mathcal{A}$ can corrupt an arbitrary subset of attribute authorities provided that adversary $\mathcal{A}$ cannot make a trivial forgery attack. These definitions are given in the full version of this paper.

5.2 Construction

The key idea of our construction of MA-ABS scheme is to share $G_{\mathsf{uid}} := \delta G_1$ as well as G_0 and G_1 among attribute authorities to generate $\delta \boldsymbol{b}^*_{t,i}$ by each authority t. Hence, G_0 and G_1 are included in tpk and $G_{\mathsf{uid}} := \delta G_1$ is shared with attribute authorities through the user's token $\mathsf{token}_{\mathsf{uid}}$.

For matrix $X := (\chi_{i,j})_{i,j=1,\ldots,N} \in \mathbb{F}_q^{N \times N}$ and element $\boldsymbol{v}$ in N-dimensional $\mathbb{V}$, $X(\boldsymbol{v})$ denotes $\sum_{i=1,j=1}^{N,N} \chi_{i,j}\phi_{i,j}(\boldsymbol{v})$ using canonical maps $\{\phi_{i,j}\}$ (Definition 2). Similarly, for matrix $(\vartheta_{i,j}) := (X^{-1})^{\mathrm{T}}$, $(X^{-1})^{\mathrm{T}}(\boldsymbol{v}) := \sum_{i=1,j=1}^{N,N} \vartheta_{i,j}\phi_{i,j}(\boldsymbol{v})$. It holds that $e(X(\boldsymbol{x}), (X^{-1})^{\mathrm{T}}(\boldsymbol{y})) = e(\boldsymbol{x}, \boldsymbol{y})$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{V}$.

Moreover, $(\mathsf{G}_{\mathsf{SIG}}, \mathsf{S}, \mathsf{V})$ is a (conventional) unforgeable signature scheme.

$\mathsf{TSetup}(1^\lambda): \quad \mathsf{param}_{\mathbb{G}} := (q, \mathbb{G}, \mathbb{G}_T, G, e) \xleftarrow{\mathsf{R}} \mathcal{G}_{\mathsf{bpg}}(1^\lambda),$

$\quad \mathsf{hk} \xleftarrow{\mathsf{R}} \mathsf{KH}_\lambda, \quad (\mathsf{verk}, \mathsf{sigk}) \xleftarrow{\mathsf{R}} \mathsf{G}_{\mathsf{SIG}}(1^\lambda) \quad N_0 := 4, \ N_{d+1} := 7, \quad \kappa, \xi \xleftarrow{\mathsf{U}} \mathbb{F}_q^\times,$

$\quad$ for $t = 0, d+1, \quad \mathsf{param}_{\mathbb{V}_t} := (q, \mathbb{V}_t, \mathbb{G}_T, \mathbb{A}_t, e) := \mathcal{G}_{\mathsf{dpvs}}(1^\lambda, N_t, \mathsf{param}_{\mathbb{G}}),$

$\qquad X_t := (\chi_{t,i,j})_{i,j} \xleftarrow{\mathsf{U}} GL(N_t, \mathbb{F}_q), \ (\vartheta_{t,i,j})_{i,j} := (X_t^{-1})^{\mathrm{T}},$

$\qquad \boldsymbol{b}_{t,i} := \kappa(\chi_{t,i,1}, \ldots, \chi_{t,i,N_t})_{\mathbb{A}_t}, \ \mathbb{B}_t := (\boldsymbol{b}_{t,1}, \ldots, \boldsymbol{b}_{t,N_t}),$

$\qquad \boldsymbol{b}^*_{t,i} := \xi(\vartheta_{t,i,1}, \ldots, \vartheta_{t,i,N_t})_{\mathbb{A}_t}, \ \mathbb{B}^*_t := (\boldsymbol{b}^*_{t,1}, \ldots, \boldsymbol{b}^*_{t,N_t}),$

$\quad G_0 := \kappa G, \quad G_1 := \xi G, \quad g_T := e(G, G)^{\kappa\xi},$

$\quad \widehat{\mathbb{B}}_0 := (\boldsymbol{b}_{0,1}, \boldsymbol{b}_{0,4}), \ \widehat{\mathbb{B}}_{d+1} := (\boldsymbol{b}_{d+1,1}, \boldsymbol{b}_{d+1,2}, \boldsymbol{b}_{d+1,7}),$

$\quad \widehat{\mathbb{B}}^*_{d+1} := (\boldsymbol{b}^*_{d+1,1}, \boldsymbol{b}^*_{d+1,2}, \boldsymbol{b}^*_{d+1,5}, \boldsymbol{b}^*_{d+1,6}),$

$\quad \mathsf{tsk} := (\boldsymbol{b}^*_{0,1}, \mathsf{sigk}),$

$\quad \mathsf{tpk} := (1^\lambda, \mathsf{hk}, \{\mathsf{param}_{\mathbb{V}_t}, \widehat{\mathbb{B}}_t\}_{t=0,d+1}, \boldsymbol{b}^*_{0,3}, \widehat{\mathbb{B}}^*_{d+1}, g_T, G_0, G_1, \mathsf{verk}),$

$\quad$ return $(\mathsf{tsk}, \ \mathsf{tpk}).$

$\mathsf{UserReg}(\mathsf{tpk}, \mathsf{tsk}, \mathsf{uid}): \ \delta \xleftarrow{\mathsf{U}} \mathbb{F}_q^\times, \quad \varphi_0, \varphi_{d+1,1,\iota}, \varphi_{d+1,2,\iota} \xleftarrow{\mathsf{U}} \mathbb{F}_q, \quad G_{\mathsf{uid}} := \delta G_1,$

$\quad \boldsymbol{k}^*_0 := (\ \delta, \ 0, \ \varphi_0, \ 0 \ )_{\mathbb{B}^*_0},$

$\quad \boldsymbol{k}^*_{d+1,1} := (\ \delta(1, \ 0), \quad 0, \ 0, \quad \varphi_{d+1,1,1}, \ \varphi_{d+1,1,2}, \quad 0 \ )_{\mathbb{B}^*_{d+1}},$

$\quad \boldsymbol{k}^*_{d+1,2} := (\ \delta(0, \ 1), \quad 0, \ 0, \quad \varphi_{d+1,2,1}, \ \varphi_{d+1,2,2}, \quad 0 \ )_{\mathbb{B}^*_{d+1}},$

$\quad \mathsf{usk}_0 := (\boldsymbol{k}^*_0, \boldsymbol{k}^*_{d+1,1}, \boldsymbol{k}^*_{d+1,2}), \quad \sigma_{\mathsf{uid}} := \mathsf{S}(\mathsf{sigk}, (\mathsf{uid}, G_{\mathsf{uid}})),$

$\quad$ return $\ \mathsf{token}_{\mathsf{uid}} := (\mathsf{uid}, G_{\mathsf{uid}}, \sigma_{\mathsf{uid}}, \mathsf{usk}_0).$

$\mathsf{ASetup}(\mathsf{tpk}): \ \boldsymbol{u}_{j,i} := (0^{i-1}, \ G_j, \ 0^{7-i})$ for $j = 0, 1; i = 1, .., 7, \ X_t \xleftarrow{\mathsf{U}} GL(7, \mathbb{F}_q),$

$\quad \mathbb{B}_t := (\boldsymbol{b}_{t,i})_{i=1,\ldots,7} := (X_t(\boldsymbol{u}_{0,1}), \ldots, X_t(\boldsymbol{u}_{0,7})),$

$\quad \mathbb{B}^*_t := (\boldsymbol{b}^*_{t,i})_{i=1,\ldots,7} := ((X_t^{-1})^{\mathrm{T}}(\boldsymbol{u}_{1,1}), \ldots, (X_t^{-1})^{\mathrm{T}}(\boldsymbol{u}_{1,7})),$

$\quad \widehat{\mathbb{B}}_t := (\boldsymbol{b}_{t,1}, \boldsymbol{b}_{t,2}, \boldsymbol{b}_{t,7}), \quad \widehat{\mathbb{B}}^*_t := (\boldsymbol{b}^*_{t,1}, \boldsymbol{b}^*_{t,2}, \boldsymbol{b}^*_{t,5}, \boldsymbol{b}^*_{t,6}),$

$\quad$ return $\ (\mathsf{ask}_t := X_t, \ \mathsf{apk}_t := (\widehat{\mathbb{B}}_t, \widehat{\mathbb{B}}^*_t)).$

$\mathsf{TokenVerify}(\mathsf{tpk}, \mathsf{uid}, \mathsf{token}_{\mathsf{uid}})$ holds iff $\mathsf{V}(\mathsf{verk}, (\mathsf{uid}, G_{\mathsf{uid}}), \sigma_{\mathsf{uid}}) = 1.$

$\mathsf{AttrGen}(\mathsf{tpk}, t, \mathsf{ask}_t, \mathsf{token}_{\mathsf{uid}}, x_t \in \mathbb{F}_q): \quad \varphi_{t,1}, \varphi_{t,2} \xleftarrow{\mathsf{U}} \mathbb{F}_q,$

$\boldsymbol{k}_t^* := (X_t^{-1})^{\mathrm{T}}((\ G_{\mathsf{uid}},\ x_t G_{\mathsf{uid}},\ \ 0,\ 0,\ \ \varphi_{t,1} G_1,\ \varphi_{t,2} G_1,\ \ 0\ )),$

that is, $\boldsymbol{k}_t^* = (\ \delta,\ \delta x_t,\ \ 0,\ 0,\ \ \varphi_{t,1},\ \varphi_{t,2},\ \ 0\ )_{\mathbb{B}_t^*}$,

return $\mathsf{usk}_t := \boldsymbol{k}_t^*$.

$\mathsf{Sig}(\mathsf{tpk},\ \mathsf{token}_{\mathsf{uid}}, \{\mathsf{apk}_t, \mathsf{usk}_t \xleftarrow{\mathsf{R}} \mathsf{AttrGen}(\mathsf{tpk}, t, \mathsf{ask}_t, \mathsf{token}_{\mathsf{uid}}, x_t)\ \mid\ (t, x_t) \in \Gamma\}, m, \mathbb{S} := (M, \rho))$ and $\mathsf{Ver}(\mathsf{tpk},\ \{\mathsf{apk}_t\}_{t=1,\ldots,d},\ m,\ \mathbb{S} := (M, \rho),\ \overrightarrow{\boldsymbol{s}}^*)$ are essentially the same as those in Section 4.2.

5.3 Security

Theorem 3. *The proposed MA-ABS scheme is perfectly private.*

Theorem 4. *The proposed MA-ABS scheme is unforgeable (adaptive-predicate unforgeable) under the DLIN assumption and the existence of collision resistant hash functions.*

For any adversary $\mathcal{A}$, there exist probabilistic machines $\mathcal{E}_1, \mathcal{E}^+_{2,h}, \mathcal{E}_{2,h+1}$ $(h = 0, \ldots, \nu_1 - 1)$, $\mathcal{E}_{3,h}, \mathcal{E}_{4,h}$ $(h = 1, \ldots, \nu_2)$, whose running times are essentially the same as that of $\mathcal{A}$, such that for any security parameter λ,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{MA\text{-}ABS,UF}}(\lambda) \le \mathsf{Adv}_{\mathcal{E}_1}^{\mathsf{DLIN}}(\lambda) + \textstyle\sum_{h=0}^{\nu_1-1} \Big(\mathsf{Adv}_{\mathcal{E}^+_{2,h}}^{\mathsf{DLIN}}(\lambda) + \mathsf{Adv}_{\mathcal{E}_{2,h+1}}^{\mathsf{DLIN}}(\lambda)\Big)$$
$$+ \textstyle\sum_{h=1}^{\nu_2} \Big(\mathsf{Adv}_{\mathcal{E}_{3,h}}^{\mathsf{DLIN}}(\lambda) + \mathsf{Adv}_{\mathcal{E}_{4,h}}^{\mathsf{H,CR}}(\lambda)\Big) + \epsilon,$$

where ν_1 is the maximum number of $\mathcal{A}$'s UserReg *queries, ν_2 is the maximum number of $\mathcal{A}$'s* AltSig *queries, and $\epsilon := ((2d + 16)\nu_1 + 18\nu_2 + 2d + 18)/q$.*

The proofs of Theorems 3 and 4 are given in the full version of this paper.

References

1. Beimel, A., Secure schemes for secret sharing and key distribution. PhD Thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)
2. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 108–125. Springer Heidelberg (2009)
3. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and noninteractive anonymous credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer Heidelberg (2008)
4. Boyen, X.: Mesh signatures. In: Naor, M. (ed.) EUROCRYPT 2007, LNCS, vol. 4515, pp. 210–227. Springer Heidelberg (2007)
5. Camenisch, J., Groß, T.: Efficient attributes for anonymous credentials. In: CCS 2008. pp.345–356. ACM (2008)
6. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optinal anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer Heidelberg (2001)
7. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer Heidelberg (2004)

8. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. In: CACM, vol. 28 (10), pp. 1030–1044. ACM (1985)
9. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT '91. LNCS, vol. 547, pp. 257–265. Springer Heidelberg (1991)
10. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer Heidelberg (2008)
11. Guo, S., Zeng, Y.: Attribute-based signature scheme, In: ISA 08, pp. 509–511. IEEE (2008)
12. Khader, D.: Attribute based group signatures, ePrint, IACR, `http://eprint.iacr.org/2007/159`.
13. Khader, D.: Attribute based group signature with revocation. ePrint, IACR, `http://eprint.iacr.org/2007/241`
14. Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption, In Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer Heidelberg (2010)
15. Lewko, A.B., Waters, B.: Decentralizing attribute-based encryption. ePrint, IACR, `http://eprint.iacr.org/2010/351`
16. Li, J., Au, M.H., Susilo, W., Xie, D., Ren. K.: Attribute-based signature and its application, In: ASIACCS 2010, pp. 60–69. ACM (2010)
17. Li, J., Kim, K.: Attribute-based ring signatures. ePrint, IACR, `http://eprint.iacr.org/2008/394`
18. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. ePrint, IACR, `http://eprint.iacr.org/2008/328`
19. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures. To appear in CT-RSA 2011. Full version is available at `http://eprint.iacr.org/2010/595`
20. Okamoto, T., Takashima, K.: Homomorphic encryption and signatures from vector decomposition. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 57–74. Springer Heidelberg (2008)
21. Okamoto, T., Takashima, K.: Hierarchical predicate encryption for inner-products. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 214–231. Springer Heidelberg (2009)
22. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer Heidelberg (2010). Full version is available at `http://eprint.iacr.org/2010/563`
23. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer Heidelberg (2001)
24. Shahandashti, S.F., Safavi-Naini, R.: Threshold attribute-based signatures and their application to anonymous credential systems. In: Preneel B. (ed.) AFRICACRYPT 2009. LNCS, vol. 5580, pp. 198–216. Springer Heidelberg (2009)
25. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO '84. LNCS, vol. 196, pp. 47–53. Springer Heidelberg (1984)
26. Waters, B.: Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer Heidelberg (2009)
27. Yang, P., Cao, Z., Dong, X.: Fuzzy identity based signature. ePrint, IACR, `http://eprint.iacr.org/2008/002`.