Short and Efficient Expressive Attribute-Based Signature in the Standard Model

Aijun Ge[1], Cheng Chen[2], Chuangui Ma[1], and Zhenfeng Zhang[2]

[1]Department of Applied Mathematics, Zhengzhou Information Science and Technology Institute, Zhengzhou, Henan 450002, China
{ geaijun2011@gmail.com, cgm20090505@163.com}
[2]State Key Laboratory of Information Security (SKLOIS), Institute of Information Engineering (IIE), Chinese Academy of Sciences (CAS), Beijing, 100190, China
{chencheng, zfzhang}@is.iscas.ac.cn

Abstract. In this paper, we present a new construction of expressive attribute-based signature scheme which is proven existentially unforgeable in the standard model for the selective adversary. In addition, this scheme can also achieve perfect privacy, which means the signature reveals nothing about the identity or attribute information about the real signer. Compared with other existing attribute-based signature schemes that can support the general access structure, the new construction provides better efficiency in terms of the computation cost and signature size.

Keywords: attribute-based signature, perfect privacy, existential unforgeability, general access structure.

1 Introduction

Attribute-based signature (ABS for short), first introduced in the full version of [10] by Maji *et al*, can be seen as the versatile and privacy enhanced variants of identity-based signature [13], in which a signer is defined by a set of attributes he owns instead of just a string representing the signer's identity. In an ABS system, users receive their secret keys from the attribute authority depending on the attributes that they hold. For the signing algorithm, a user can use his private key and a signing predicate (which must be satisfied by his attributes) to sign on a message. Everybody can convince that some user who holds a set of attributes satisfying the signing predicate has endorsed the message. Particularly, the signature releases no other identity or attribute information about the real signer. This is also known as the perfect privacy. Collusion resistance is also the essential security property of ABS schemes, which means that a signature cannot be computed by a signer whose attributes don't satisfy a given predicate, even multi-users collude to pool their attribute together. ABS has found many important application scenarios requiring both privacy and authentication, such as attribute-based messaging, attribute-based authentication, leaking secrets and trust negotiation (see [10] for details).

Since the first ABS scheme was introduce by Maji *et al* [10], ABS has received much attention, and a number of ABS schemes have been proposed in recent years. According to the structure of access policy, these schemes can be divided into two categories: ABS for threshold structures (such as [4,8,14,7,6]) and ABS for general access structures (such as [10,11,3,12]). Specially, ABS with full security for non-monotone predicate and multi-authority setting are also proposed in [11,12]. The main disadvantage of their schemes [11,12] is the long signature size (which comprise $7n + 11$ group elements in [11] and $13n$ group elements in [12], where n is the number of attributes involved in the signing predicate), and hence their scheme has lower communication efficiency in the sense of signature size.

One of the major issues in ABS is to save bandwidth, and in particular to get signatures of constant size, regardless of the number of involved attributes. Note that ABS schemes with constant size signatures for threshold predicates have been proposed in two independent papers [4,6]. How to construct a constant size ABS scheme for general access structures is still a very challenging open problem. The goal of this paper is to construct the "almost optimally efficient" ABS for expressive signing predicates, though its signatures length still cannot achieve the constant size.

Our Contributions. Motivated by the recent work of [11,12], we give a new construction of ABS scheme based on Waters ciphertext-policy attribute-based encryption schemes [18]. Our concrete scheme is provably secure against selective attacks under the assumption of computation q-Diffie-Hellman Exponent (q-DHE) problem, which is modified from the decisional q-Bilinear Diffie-Hellman Exponent (q-BDHE) problem introduced by Boneh *et al.* in [2]. Compared with other ABS schemes that can support general access structures, our construction provides better efficiency in terms of the computation cost and communication cost. What is more, the signature size in our construction is even shorter than Maji *et al.*'s [10] most efficient construction which is only proven secure in the generic group model. It is well known that the generic group model is not a standard model in the security proof, while our scheme can be proved secure in the standard model (without using the random oracle or the generic group heuristic).

Organization. The rest of this paper is organized as follows. In the next section, we briefly review some preliminaries, including the bilinear maps, hardness assumption, access structure and the syntax definition of ABS scheme. The security model is given in Section 3. We present the concrete ABS scheme in Section 4. Further, we compare the efficiency and security of our ABS scheme with other existing ABS schemes in Section 5. Finally, conclusions will be made in Section 6.

2 Preliminaries

2.1 Bilinear Maps

Let $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ be cyclic multiplicative groups of prime order p. g is a generator of $\mathbb{G}_1$ and h is a generator of $\mathbb{G}_2$. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ has the following properties:

- **Bilinearity**: $e(g^a, h^b) = e(g, h)^{ab}$ for any $a, b \in \mathbb{Z}_p^*$, .
- **Non-degeneracy**: $e(g, h) \neq 1$.
- **Computability**: There's an efficient algorithm to compute $e(u, v)$ for any $u \in \mathbb{G}_1$ and $v \in \mathbb{G}_2$.

2.2 Hardness Assumption

Definition 1. *(q-Diffie-Hellman Exponentiation, q-DHE) We say the (t, ε) q-DHE assumption holds in a group $\mathbb{G}$, if there is no probabilistic polynomial time adversary who is able to compute $g^{a^{q+1}}$ just given $(g, g^a, g^{a^2}, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}} \in \mathbb{G}^{2q})$ running in time at most t with probability at least ε, where $a \in \mathbb{Z}_p$ and $g \in \mathbb{G}$ are chosen independently and uniformly.*

2.3 Access Structure and Linear Secret Sharing Scheme

Definition 2. *(Access structure[1]) Let $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ be a set of parties, and a collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}}$ is called monotone if $\forall B, C \subseteq 2^{\{P_1, P_2, ..., P_n\}}$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, ..., P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets.*

Definition 3. *(Linear secret sharing scheme [1], LSSS) A secret sharing scheme Π over a set of parties $\mathcal{P} = \{P_1, P_2, ..., P_n\}$ is called LSSS if*

1. *The share for each party forms a vector over $\mathbb{Z}_p$;*
2. *There exists a matrix M with ℓ rows and k columns called the share generating matrix for Π. For $i = 1, ..., \ell$, we let the function ρ define the party labeling the row i of M as $\rho(i)$. When we consider the column vector $\overrightarrow{v} = (v_1, v_2, ..., v_k)^T$, where $v_1 = s \in \mathbb{Z}_p$ is the secret to be shared, and $v_2, ..., v_k \in \mathbb{Z}_p$ are chosen randomly, then $M\overrightarrow{v}$ is the vector of ℓ shares of the secret s according to Π. The share $\lambda_i = (M\boldsymbol{v})_i$ belongs to the party $\rho(i)$.*

Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be an authorized set, and $I \subseteq \{1, 2, ..., \ell\}$ is defined as $I = \{i : \rho(i) \in S\}$. If $\{\lambda_i\}_{i \in I}$ are valid shares of any secret s according to Π, there exist constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ such that $\sum_{i \in I} \lambda_i w_i = s$. It is shown in [1] that these constants $\{w_i\}_{i \in I}$ can be found in polynomial time .

Using standard techniques [1], one can convert any monotonic boolean formulas into an LSSS representation. An access tree of ℓ nodes will result in an LSSS matrix of ℓ rows. It is shown in [9] that, one can convert any access tree into an LSSS matrix. The signing predicate will used in our ABS scheme is also the LSSS access structure.

2.4 Syntax of ABS Scheme

According to [10], an ABS scheme consists of four algorithms: **Setup**, **Extract**, **Sign** and **Verify**. For a fixed security parameter λ, these algorithms work as follows:

- **Setup**(λ): The Setup algorithm takes the security parameter λ as input, and it returns some public parameters $params$ and the master secret key msk. Here we assume that the universe of attributes $\mathbb{U}$ is also contained in the public parameters $params$.
- **Extract**$(msk, params, S)$: The Extract algorithm takes input the master secret key msk, the public parameters $params$ and a user's attribute set $S \subseteq \mathbb{U}$, and the attribute authority computes the attribute private key SK_S as the algorithm's output.
- **Sign**$(m, params, \Upsilon, SK_S)$: The Sign algorithm takes input a signing message m, the public parameters $params$, a signing predicate Υ and a user's private key SK_S with the attribute set S satisfying the predicate Υ (we say that an attribute set S satisfies the predicate Υ if and only if $\Upsilon(S) = 1$), and outputs a valid signature σ.
- **Verify**$(params, \Upsilon, m, \sigma)$: The Verify algorithm takes the public parameters $params$, the signing predicate Υ, the message m and its signature σ as input, and outputs a boolean value either "valid" or "invalid".

Correctness. For any correctly generated signature on the message m by a signer with attributes S satisfying the claim-predicate Υ, we have

$$\textbf{Verify}(params, \Upsilon, m, \textbf{Sign}(m, params, \Upsilon, SK_S)) = valid.$$

3 Security Models

Two security properties: unforgeability and perfect privacy must be satisfied by an ABS scheme. The definition of unforgeability is based on the idea that an adversary can not generate a valid signature if his attribute set S does not satisfy a given predicate Υ. For the perfect privacy, the signature reveals nothing information about the identity or attributes of the signer. Our security model here combines the model proposed by Maji *et al.* [10] and Li *et al.*[7].

3.1 Unforgeability

Our ABS scheme is existentially unforgeable against selective predicate attacks, which is weaker than the fully chosen predicate attacks. The difference is that the selective predicate adversary has to choose the challenge predicate before the setup phase. Note that this weaker model has also been used in other attribute-based signature schemes [8,14,7,4,6]. We now give the formal security definition by the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

- **Initial Phase**: The adversary $\mathcal{A}$ gives a challenge predicate Υ^* to the challenger $\mathcal{C}$, which will be used in the forgery signature.
- **Setup Phase**: After receiving the challenge predicate Υ^*, the challenger $\mathcal{C}$ chooses a security parameter λ and runs the Setup algorithm to generate the master secret key msk and public parameters $params$. $\mathcal{C}$ gives public parameters $params$ to the adversary $\mathcal{A}$, while keeps the master secret key msk secret.
- **Queries Phase**: $\mathcal{A}$ can adaptively query Extraction Oracle and Signing Oracle for a polynomially bounded times, and $\mathcal{C}$ answers these queries with the master secret key msk:
  - Extraction Oracle: $\mathcal{A}$ can request the private key SK_S for any attribute set $S \subseteq \mathbb{U}$.
  - Signing Oracle: $\mathcal{A}$ can request the signature for any message m and any predicate Υ.
- **Forgery Phase**: Finally, the adversary $\mathcal{A}$ outputs a signature σ^* on a message m^* with respect to the challenge predicate Υ^*. We say the adversary $\mathcal{A}$ wins the game if
  1. $\mathcal{A}$ has not made Extraction Oracle for any attribute set S^* satisfying Υ^*;
  2. (m^*, Υ^*) has not been queried to the Signing Oracle;
  3. σ^* is a valid signature on the message m^* and predicate Υ^*, which means that $(m^*, \sigma^*, \Upsilon^*)$ can pass the Verify algorithm.

The advantage $Adv^{sP-CMA-EUF}_{ABS,A}$ is defined as the probability that $\mathcal{A}$ wins in the above game.

Definition 4. *An adversary $(t, q_K, q_S, \varepsilon)$ breaks an ABS scheme if the adversary $\mathcal{A}$ runs in time at most t, and makes at most q_K and q_S times Extraction Oracle queries and Signing Oracle queries, while the advantage $Adv^{sP-CMA-EUF}_{ABS,A}$ is at least ε. An ABS scheme is $(t, q_K, q_S, \varepsilon)$ existentially unforgeable if there is no forger that can $(t, q_K, q_S, \varepsilon)$ break it.*

3.2 Perfect Privacy

Definition 5. *An ABS scheme satisfies the property of perfect privacy if for any two attribute sets S_1 and S_2, any message m, any signature σ on any predicate Υ with $\Upsilon(S_1) = \Upsilon(S_2) = 1$, any adversary $\mathcal{A}$ cannot identify which attribute set S_1 or S_2 is used to generate the signature σ better than random guessing.*

If perfect privacy holds, then any adversary does not know which set of attributes of signing key was used to generate the signature. This holds even if the adversary has unbounded computational power and has access to the signer's private keys, i.e., the signature is simply independent of everything except the message and the predicate.

4 New Construction of ABS Scheme

In this section, we present our ABS scheme with short signature size, which is proven selective secure under the q-DHE assumption (Definition 1) in the standard model.

- **Setup:** Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be two cyclic multiplication groups of prime order p, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ is an efficient bilinear map. Note that this construction can also work on asymmetric pairing groups, where $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ and $\mathbb{G}_1 \neq \mathbb{G}_2$. The universal set of attributes $\mathbb{U}$ ($|\mathbb{U}| = U, U \in \mathbb{Z}_p$) used in this system is public known. First, select a random generator $g \in \mathbb{G}_1$, two exponents $\alpha, a \in \mathbb{Z}_p^*$, and compute $h = g^a$,$Z = e(g,g)^\alpha$. Next, $h_1, h_2, ..., h_U, u_0, u_1, u_2, ..., u_n$ are randomly chosen from $\mathbb{G}_1$. We use $u_0, u_1, u_2, ..., u_n$ as the Waters hash function [17], and n is the size of a message used in this system. The master key is g^α, and the public parameters are $params = (g, h, Z, h_1, h_2, ..., h_U, u_0, u_1, u_2, ..., u_n)$.
- **Extract:** The private key for a user with attributes $S \subseteq \mathbb{U}$ is computed as follows:
  1. The attribute authority randomly selects $t \in \mathbb{Z}_p$, then computes $K = g^\alpha h^t$, $L = g^t$;
  2. For each attribute $x \in S$, the attribute authority computes $K_x = h_x^t$;
  3. Finally, the attribute authority outputs $SK_S = (K, L, \{K_x\}_{x \in S})$ as the private key for attributes S.
- **Sign:** Let the signing predicate Υ can be represented by an LSSS access structure (M, ρ), i.e., M is an $\ell \times k$ matrix, and ρ is an injective function that maps each row of the matrix M to an attribute. The signature for a message $m = (\mu_1, \mu_2, ..., \mu_n) \in \{0,1\}^n$ is computed as follows:
  1. The signer should first blind his private key SK_S in the following way: The signer chooses a random $t' \in \mathbb{Z}_p$, and sets $K' = Kh^{t'} = g^\alpha h^{t+t'}$, $L' = Lg^{t'} = g^{t+t'}$, $\forall x \in S : K'_x = K_x h_x^{t'} = h_x^{(t+t')}$. Then the signer's new private key $SK'_S = (K', L', \{K'_x\}_{x \in S})$.
  2. As the signer's attributes S should satisfy the signing predicate $\Upsilon(\Upsilon(S) = 1)$, the signer can find vector $\overrightarrow{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_\ell)$ that satisfies $\overrightarrow{\alpha} M = (1, 0, ..., 0)$ according to Definition 3. In addition, the signer can find another vector $\overrightarrow{\beta} = (\beta_1, \beta_2, ..., \beta_\ell)$ that satisfies $\overrightarrow{\beta} M = (0, 0, ..., 0)$. For the existence of $\overrightarrow{\beta}$, we have a discussion in Section 5.2.
  3. For each $i \in [1, \ell]$, the signer first computes $s_i = (L')^{\alpha_i} g^{\beta_i}$ and sets $y = \prod_{i=1}^{\ell} ((K'_{\rho(i)})^{\alpha_i} (h_{\rho(i)})^{\beta_i})$. Then, the signer randomly chooses $r \in \mathbb{Z}_p$ and computes: $\sigma_1 = yK'(u_0 \prod_{j=1}^{n} (u_j)^{\mu_j})^r$, $\sigma_2 = g^r$.
     Note: The signer may not have the key $K'_{\rho(i)}$ for every attribute $\rho(i)$ in the computation of y. However, in this case, $\alpha_i = 0$, and so the value $K'_{\rho(i)}$ is not needed.
  4. Finally, the signer outputs the signature for the message m and the signing predicate $\Upsilon(M, \rho)$: $\sigma = (s_1, ..., s_\ell, \sigma_1, \sigma_2)$.

– **Verify:** Given a signature $\sigma = (s_1, ...s_\ell, \sigma_1, \sigma_2)$ of a message $m = (\mu_1, \mu_2, ..., \mu_n) \in \{0,1\}^n$ for the predicate Υ(corresponding to $(M_{\ell \times k}, \rho)$), firstly, the verifier chooses randomly $v_2, ..., v_k$ from $\mathbb{Z}_p$, sets $\overrightarrow{v} = (v_1 = 1, v_2, ..., v_k)$ and computes $\lambda_i = \sum_{j=1}^{k} v_j M_{i,j}$, where M_i is the vector corresponding to the ith row of matrix M. The verifier accepts the signature if and only if the following equation holds, otherwise rejects it.

$$e(g, \sigma_1) \stackrel{?}{=} Z e((u_0 \prod_{j=1}^{n} (u_j)^{\mu_j}), \sigma_2) \prod_{i=1}^{\ell} e(h^{\lambda_i} h_{\rho(i)}, s_i).$$

5 Security Analysis

The correctness of this ABS algorithm follows from the correctness of Waters CP-ABE scheme [18], and is shown in Appendix A.

5.1 Existential Unforgeability

Theorem 1. *The new ABS scheme is existentially unforgeable under the selective predicate attack, assuming that the q-DHE assumption holds in* $\mathbb{G}_1$.

Proof. Suppose there exists an adversary $\mathcal{A}$ with non-negligible advantage ε against our ABS scheme, then we can construct a probability polynomial time algorithm that can solve the q-DHE problem. We define the game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

Init: The challenger $\mathcal{C}$ is given a q-DHE challenge $(g, g^a, g^{a^2}, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}})$ and is asked to compute $g^{a^{q+1}}$. The game begins with $\mathcal{A}$ sends a challenge predicate $\Upsilon^*(M^*, \rho^*)$, where the size of the the challenge matrix M^* is $\ell^* \times k^*$ with $k^* \leqslant q$.

Setup: $\mathcal{C}$ first defines the parameters $h_1, h_2, ..., h_U$ as follows. For each attribute x in this system, $\mathcal{C}$ chooses a random value $z_x \in \mathbb{Z}_p$. If an attribute x appeared in the challenge predicate $\Upsilon^*(M^*, \rho^*)$, i.e., there is an $i \in [1, \ell^*]$ such that $\rho^*(i) = x$, $\mathcal{C}$ then lets $h_x = g^{z_x} g^{-aM^*_{i,1}} g^{-a^2 M^*_{i,2}} ... g^{-a^{k^*} M^*_{i,k^*}}$, otherwise $h_x = g^{z_x}$. Further more, $\mathcal{C}$ chooses $\alpha', b_j \in \mathbb{Z}_p, a_j \in \{-1, 0, 1\}$ randomly for each $0 \leqslant j \leqslant n$, and sets $h = g^a$, $Z = e(g,g)^\alpha = e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha'}$, $u_j = g_1^{a_j} g^{b_j} (0 \leqslant j \leqslant n)$ with $g_1 = g^{a^q}$. Note that the master secret key $g^\alpha = g^{a^{q+1}} \cdot g^{\alpha'}$ is unknown to $\mathcal{C}$. For convenience, let $a(m) = a_0 + \sum_{i=1}^{n} a_i m_i$, $b(m) = b_0 + \sum_{i=1}^{n} b_i m_i$ just like Waters hash function [17].

Queries: The adversary $\mathcal{A}$ can adaptively request the following oracles for polynomial times, and $\mathcal{C}$ answers these queries in the following way:

– Extraction Oracles: Suppose the adversary $\mathcal{A}$ query the private key for a user's attribute set $S \subseteq \mathbb{U}$ with the restriction that S does not satisfy M^*. The challenger $\mathcal{C}$ randomly chooses $t_1 \in \mathbb{Z}_p$, and it finds a vector $\overrightarrow{w} = (w_1, ..., w_{k^*}) \in \mathbb{Z}_p^{k^*}$ such that $w_1 = -1$ and for all i that $\rho^*(i) \in S$, $\overrightarrow{w} \cdot M^*_i =$

0. By implicitly defining $t = t_1 + w_1 a^q + w_2 a^{q-1} + ... + w_{k^*} a^{q-k^*+1}$, $\mathcal{C}$ outputs the private key $SK_S = (K, L, \{K_x\}_{x \in S})$ as follows:

$$K = g^{\alpha} h^t = g^{\alpha'} g^{at_1} \prod_{i=2,...,k^*} (g^{a^{q+2-i}})^{w_i}$$

$$L = g^t = g^{t_1} \prod_{i=1,...,k^*} (g^{a^{q+1-i}})^{w_i}$$

$K_x = L^{z_x} \prod_{j=1}^{k^*} (g^{a^j t_1} \prod_{k=1,k\neq j}^{k^*} (g^{a^{q+1+j-k}})^{w_k})^{-M^*_{i,j}}$ for $\rho^*(i) = x \in S$, and $K_x = (h_x)^t = L^{z_x}$ for the attribute x that does not appear in the challenge predicate (i.e., there is no i such that $\rho^*(i) = x$).

– Signing Oracles: Consider a query for a signature of the message $m = (\mu_1, \mu_2, ..., \mu_n) \in \{0,1\}^n$ on the predicate $\Upsilon(M_{\ell \times k}, \rho)$ with attributes S that satisfies $\Upsilon(S) = 1$. The challenger $\mathcal{C}$ will compute a valid signature in the following way:
  1. If S does not satisfy the challenge predicate Υ^*, $\mathcal{C}$ can get the private key of S by querying the Extraction Oracles, and then generates a valid signature trivially using the private key.
  2. If S does satisfy the challenge predicate Υ^*, that is $\Upsilon^*(S) = 1$. $\mathcal{C}$ first checks $a(m) = a_0 + \sum_{i=1}^{n} a_i \mu_i$ equals 0 or not. If $a(m) = 0$, $\mathcal{C}$ outputs "failure" and stops this game. Otherwise, when $a(m) \neq 0$, $\mathcal{C}$ can compute $\overrightarrow{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_\ell)$ and $\overrightarrow{\beta} = (\beta_1, \beta_2, ..., \beta_\ell)$ with $\overrightarrow{\alpha} M = (1, 0, ..., 0)$, $\overrightarrow{\beta} M = (0, 0, ..., 0)$ as $\Upsilon(S) = 1$. $\mathcal{C}$ randomly chooses $r', t' \in \mathbb{Z}_p$ and assigns $r = r' - (a/a(m))$, and computes a signature $\sigma = (s_1, ...s_\ell, \sigma_1, \sigma_2)$, where:

$$s_i = g^{\alpha_i t' + \beta_i} (i = 1, 2..., \ell),$$

$$y = \prod_{i=1}^{\ell} (h_{\rho(i)})^{\alpha_i \cdot t' + \beta_i},$$

$$\sigma_1 = y g^{\alpha + at'} (g_1^{a(m)} g^{b(m)})^r)$$
$$= y (g^{\alpha'} (g^a)^{t'} (g^{a^q})^{r' a(m)} g^{b(m) \cdot (r' - a/a(m))}),$$

$$\sigma_2 = g^{r' - (a/a(m))} = g^{r'} (g^a)^{-(1/a(m))}.$$

Forgery: Finally, the adversary $\mathcal{A}$ outputs a forged signature $\sigma^* = (s_1^*, ...s_\ell^*, \sigma_1^*, \sigma_2^*)$ of a message $m^* = (\mu_1^*, \mu_2^*, ..., \mu_n^*) \in \{0,1\}^n$ for the challenge predicate Υ^* (the corresponding monotone span program is $(M^*_{\ell^* \times k^*}, \rho^*)$). If $a(m^*) \neq 0$, $\mathcal{C}$ will abort. Otherwise, $\mathcal{C}$ can give the solution to the q-DHE problem. $\mathcal{C}$ implicitly sets $\overrightarrow{v} = (s = 1, a, a^2 ..., a^{k^*-1})$, and computes $\lambda_i = \boldsymbol{v} \cdot M_i^* = \sum_{j=1}^{\ell^*} M^*_{i,j} a^{j-1}$. As $h_{\rho*(i)} = g^{z_{\rho*(i)}} g^{-aM^*_{i,1}} g^{-a^2 M^*_{i,2}} ... g^{-a^{k^*} M^*_{i,k^*}} = g^{z_{\rho*(i)}} g^{-a\lambda_i}$, $\mathcal{C}$ can compute $\sigma_1 = g^{\alpha} \sigma_2^{b(m^*)} (\prod_{i=1}^{\ell^*} s_i^{z_{\rho*(i)}})$. According to $g^{\alpha} = g^{a^{q+1}} \cdot g^{\alpha'}$ for the above definition, $\mathcal{C}$ outputs

$$g^{a^{q+1}} = \sigma_1 (\sigma_2)^{-b(m*)} g^{-\alpha'} \prod_{i=1}^{\ell^*} s_i^{-z_{\rho*(i)}}$$

as the solution to the the q-DHE problem instance, which contradicts with q-DHE assumption.

Probability: For the simulation to complete without aborting, we require the following conditions fulfilled:

1. For each m_i to be queried in the Signing Oracle queries, $a(m_i) \neq 0$;
2. For the forged message m^*, $a(m^*) = 0$.

Using the technique of $(1, q_S, 0, P)$ programmable hash function [15], we get the probability that $\mathcal{C}$ does not abort as $P = O(\frac{1}{q_S\sqrt{n}})$, where q_S is the maximum number of Signing Oracle queries the adversary $\mathcal{A}$ can make. Therefore, the probability of solving q-DHE problem as $\varepsilon' \geqslant \varepsilon \cdot P$, if the adversary $\mathcal{A}$ can succeed breaking our ABS scheme with probability ε. □

5.2 Perfect Privacy

Theorem 2. *The ABS scheme in Section 4 can achieve perfect privacy.*

Proof. First we point that the role of the vector $\overrightarrow{\beta} = (\beta_1, \beta_2, ..., \beta_\ell)$ in the process of computing the signature is to hide the real attributes the signer used to sign the message. For any predicate Υ (The corresponding LSSS access structure is $(M_{\ell\times k}, \rho)$), if $rank(M) < \ell$, it is obvious that there exist polynomial numbers of $\overrightarrow{\beta}$ that satisfies $\overrightarrow{\beta}M = (0, 0, ..., 0)$. For a simple example, M could be $(1, 1, 1, 1)^T$ (a column vector, which is the transpose of $(1, 1, 1, 1)$ for the policy: A_1 or A_2 or A_3 or A_4 (following the way of [9] in the construction of M). In that case, $\overrightarrow{\beta} = (\beta_1, \beta_2, \beta_3, \beta_4)$ could be $(1, -1, 1, -1)$ or $(1, 1, 1, -3)$ or any other values that satisfies $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 0$. On the other hand, if the matrix M is full rank, then there is only one $\overrightarrow{\beta} = (0, 0, ..., 0)$ that satisfies $\overrightarrow{\beta}M = (0, 0, ..., 0)$. But in this case, as $\overrightarrow{\alpha}M = (1, 0, ..., 0)$, $\overrightarrow{\alpha} = (1, 1, ..., 1)$, which means the signing predicate are limited to conjunction or (n,n)-threshold predicate, and there is no attribute privacy at all. So here, we just consider the case of $rank(M) < \ell$.

The perfect privacy game begins with the challenger running Setup algorithm to get the public parameters *params* and the master key g^α. The challenger also gives the adversary the public parameters *params* as well as the master key g^α. After these interactions, the adversary outputs a challenge predicate Υ and two attributes S_1 and S_2 with $\Upsilon(S_1) = \Upsilon(S_2) = 1$. Assume the challenger has generated the private keys (which are also given to the adversary) as

$$SK_{S_1} = (g^\alpha h^{t_1}, g^{t_1}, \{(h_x)^{t_1}\}_{x\in S_1}),$$

$$SK_{S_2} = (g^\alpha h^{t_2}, g^{t_2}, \{(h_x)^{t_2}\}_{x\in S_2}).$$

The adversary outputs a challenge message $m = (\mu_1, \mu_2, ..., \mu_n) \in \{0, 1\}^n$ and asks the challenger to generate a signature on the message m and predicate Υ with the private key SK_S either from SK_{S_1} or SK_{S_2}. The challenger chooses a random bit $b \in \{1, 2\}$, and computes a signature $\sigma^* = (s_1^b, ..., s_\ell^b, \sigma_1^b, \sigma_2^b)$ by running Sign algorithm with the private key SK_{S_b}

As the challenger can generate a valid signature by using SK_{S_1} or SK_{S_2}, and for our purpose here, we just have to show that the distribution of a signature created by using SK_{S_1} or SK_{S_2} is identical. We will show that an ABS created by using SK_{S_1} can also be generated by using SK_{S_2}:

Using the private key SK_{S_1}, the challenge signature $\sigma^* = (s_1^1, ..., s_\ell^1, \sigma_1^1, \sigma_2^1)$ is in the form:

$$\begin{aligned}
&\{s_i^1 = g^{\alpha_i^1 t + \beta_i^1}\}_{i=1,2,...,\ell},\\
&y^1 = \prod_{i=1}^{\ell} (h_{\rho(i)})^{\alpha_i^1 t + \beta_i^1},\\
&\sigma_1^1 = y^1 g^{\alpha} h^t (u_0 \prod\nolimits_{j=1}^{n} (u_j)^{\mu_j})^{r^1}),\\
&\sigma_2^1 = g^{r^1},
\end{aligned}$$

where $t = t_1 + t_1'$(t_1' and r^1 are randomly chosen by the challenger). The vector $\overrightarrow{\alpha^1} = (\alpha_1^1, \alpha_2^1, ..., \alpha_\ell^1)$ satisfying $\overrightarrow{\alpha^1} M = (1, 0, ..., 0)$ according to attributes S_1, and the vector $\overrightarrow{\beta^1} = (\beta_1^1, \beta_2^1, ..., \beta_\ell^1)$ satisfying $\overrightarrow{\beta^1} M = (0, 0, ..., 0)$ are chosen randomly by the challenger.

For the private key SK_{S_2}, there is another vector $\overrightarrow{\alpha^2} = (\alpha_1^2, \alpha_2^2, ..., \alpha_\ell^2)$ satisfying $\overrightarrow{\alpha^2} M = (1, 0, ..., 0)$. Note that $t_2 + t_2' = t = t_1 + t_1'$, $r^2 = r^1$ where t_2' and r^2 are also randomly chosen by the challenger.

In addition, we set $\overrightarrow{\beta^2} = (\beta_1^2, \beta_2^2, ..., \beta_\ell^2)$ and $\beta_i^2 = (\alpha_i^1 - \alpha_i^2)t + \beta_i^1$ for each $i \in [1, \ell]$. We can check that

$$\begin{aligned}
\overrightarrow{\beta^2} M &= \sum\nolimits_{i=1}^{\ell} \beta_i^2 M_i\\
&= \sum\nolimits_{i=1}^{\ell} (\alpha_i^1 - \alpha_i^2) t M_i + \sum\nolimits_{i=1}^{\ell} \beta_i^1 M_i\\
&= t(\overrightarrow{\alpha^1} M) - t(\overrightarrow{\alpha^2} M) + (\overrightarrow{\beta^1} M)\\
&= (t, 0, ..., 0) - (t, 0, ..., 0) + (0, 0, ..., 0)\\
&= (0, 0, ..., 0)
\end{aligned}$$

So, by choosing proper random t_2', r^2 and $\overrightarrow{\beta^2}$, the challenger can generate the same valid signature $\sigma^* = (s_1^1, ..., s_\ell^1, \sigma_1^1, \sigma_2^1)$ from the private key SK_{S_2}. By using the similar proof, one can also get the following result: if a signature is generated by the private key SK_{S_2}, it can also be generated from the private key SK_{S_1}. From the above proof, we have shown that the ABS scheme we proposed in Section 4 can achieve perfect privacy. □

5.3 Efficiency

In this section, we compare our scheme with other existing ABS schemes that can support the general access structure in the literature. The instantiations

1,2 in [10] and the schemes in [3] can be seen as the general constructions, and these schemes are very inefficient (indicated by the comparison in [11]), as they will employ the Groth-Sahai non-interactive proof [5], so these schemes are not included in the Table 1. We refer interested readers to [11] for the complexity of ABS scheme using the Groth-Sahai proof [5]. In addition, the size of private key in our scheme equals the instantiation 3 in [10], while the public parameters in our scheme are quite similar as in [10], which are both shorter than the scheme in [11] and [12], so we have not discussed in the Table 1.

Let ℓ and k represent the size of the underlying access structure matrix $M_{\ell \times k}$ for a signing predicate. According to the Table 1, the signature size of our scheme is $\ell + 2$ group elements, and the complexity is only $\ell + 3$ paring operations (Here we only consider the complex pairing operations in the Table 1), so our ABS scheme has the shortest signature size and the best computation efficiency.

Table 1. Comparison with other ABS schemes

Schemes	MPR11 [10]	OT11 [11]	OT12 [12]	Ours
Signature Size	$\ell + k + 2$	$7\ell + 11$	13ℓ	$\ell + 2$
Complexity	$k\ell + k + 3$	$7\ell + 15$	13ℓ	$\ell + 3$
Security	full	full	full	selective
Model	generic group	standard	random oracle	standard
Predicate	monotone	non-monotone	non-monotone	monotone
Multi-authority	No	Yes	Yes	No

Note that ABS schemes [4,6] with constant size signatures could also be extended to support some other more expressive kinds of monotone predicates (such as hierarchical threshold predicates [16], but not the fully expressive access structure), in which case the signature size is no longer constant. Our scheme is more efficient compared with ABS schemes [4,6] for the expressive access structure such as the CNF or DNF form.

6 Conclusion

In this paper, we give a new construction of ABS scheme based on Waters CP-ABE scheme [18]. Under the q-DHE assumption, this ABS scheme is existentially unforgeable against selective predicate attacks in the standard model. Compared with other ABS schemes that support the general signing policies, our scheme can achieve the best efficiency in terms of the signature size and computation costs, at the expense of a weaker security. We remark that the construction of a fully secure short and efficient expressive attribute-based signature, which is secure against fully adaptive adversaries in the standard model, will be our future work.

References

1. Amos Beimel. Secure schemes for secret sharing and key distribution. PhD thesis, Isral institute of technology, Technion, Haifa, Israel, 1996.
2. Boneh D, Boyen X, Goh E. Hierarchical identity-based encryption with constant size ciphertext. In *EUROCRYPT 2005*, *LNCS 3494*, Springer-Verlag, 2005, pp.440-456.
3. Escala A, Herranz J, Morillo Paz. Revocable attribute-based signatures with adaptive security in the standard model. In *AFRICACRYPT 2011*, *LNCS 6737*, Springer-Verlag, 2011, pp.224-241.
4. Ge A, Ma C, Zhang Z. Attribute-based signature scheme with constant size signature in the standard model. In *IET Information Security*, 2012, 6(2):1-8.
5. Groth J, Sahai A. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, *LNCS 4965*, Springer-Verlag, 2008, pp.415-432.
6. Herranz J, Laguillaumie F, Libert B, Rafols C. Short attribute-based signature for threshold predicates. In *CT-RSA 2012*, *LNCS 7178*, Springer-Verlag, 2012, pp.51-67.
7. Li J, Au M, Susio W, Xie D, Ren R. Attribute-based signature and its applications. In *ASIACCS 2010*, ACM, 2010, pp.60-69.
8. Li J, Kim K. Hidden attribute-based signatures without anonymity revocation, *Information Sciences*, 2010, 180(1), pp.1681-1689.
9. Lewko A, Waters B. Decentralizing attribute-based encryption. In *EUROCRYPT 2011*, *LNCS 6632*, Springer-Verlag, 2011, pp.568-588.
10. Maji H, Prabhakaran M, Rosulek M. Attribute-based signature: Achieving attribute privacy and collusion-resistance. In *CT-RSA 2011*, *LNCS 6558*, Springer-Verlag, 2011, pp.376-392. The first version available at http://eprint.iacr.org/2008/328.
11. Okamoto T, Takashima K. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *PKC 2011*, *LNCS 6571*, Springer-Verlag, 2011, pp.35-52.
12. Okamoto T, Takashima K. Decentralized attribute-based signatures. *Cryptology ePrint Archive*, Report 2011/701, 2011. http://eprint.iacr.org/2011/701
13. Shamir A. Identity-based cryptosystems and signature schemes. In *CRYPTO 1984*, *LNCS 196*, Springer-Verlag, 1984, pp.47-53.
14. Shahandashti S, Safavi-Naini R. Threshold attribute-based signature and their application to anonymous credential systems. In *AFRICACRYPT 2009*, *LNCS 5580*, Springer-Verlag, 2009, pp.198-216.
15. Schage S, Schwenk J. A CDH-based ring signature scheme with short signatures and public keys. In *FC 2010*, *LNCS 6052*, Springer-Verlag, 2010, pp.129-142.
16. Tassa T. Hierarchical threshold secret sharing. Journal of Cryptology, 20(2), pp. 237-264, 2007.
17. Waters B. Efficient identity-based encryption without random oracle. In *EUROCRYPT 2005*, *LNCS 3494*, Springer-Verlag, 2005, pp.114-127.
18. Waters B. Ciphertext policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *PKC 2011*, *LNCS 6571*, Springer-Verlag, 2011, pp.53-70.

A Correctness

If $\sigma = (s_1, ..., s_\ell, \sigma_1, \sigma_2)$ is a valid signature of the message $m = (\mu_1, \mu_2, ..., \mu_n) \in \{0,1\}^n$ for the predicate Υ (corresponding to $(M_{\ell \times k}, \rho)$), then

$$\begin{aligned}
\sigma_1 &= yK'(u_0 \prod_{j=1}^{n} (u_j)^{\mu_j})^r) \\
&= K' \prod_{i=1}^{\ell} ((K'_{\rho(i)})^{\alpha_i} (h_{\rho(i)})^{\beta_i}) (u_0 \prod_{j=1}^{n} (u_j)^{\mu_j})^r \\
&= g^{\alpha} h^t \prod_{i=1}^{\ell} (h_{\rho(i)}^{\alpha_i t + \beta_i}) (u_0 \prod_{j=1}^{n} (u_j)^{\mu_j})^r
\end{aligned}$$

$$\begin{aligned}
e(g, \sigma_1) &= e(g, g^{\alpha} h^t \prod_{i=1}^{\ell} (h_{\rho(i)}^{\alpha_i t + \beta_i}) (u_0 \prod_{j=1}^{n} (u_j)^{\mu_j})^r) \\
&= Ze(u_0 \prod_{j=1}^{n} (u_j)^{\mu_j}, g^r) e(g, h^t \prod_{i=1}^{\ell} (h_{\rho(i)}^{\alpha_i t + \beta_i})) \\
&= Ze(u_0 \prod_{j=1}^{n} (u_j)^{\mu_j}, \sigma_2) e(g, h^t) \prod_{i=1}^{\ell} e(g^{\alpha_i t + \beta_i}, h_{\rho(i)}) \\
&= Ze(u_0 \prod_{j=1}^{n} (u_j)^{\mu_j}, \sigma_2) \prod_{i=1}^{\ell} e(s_i, h^{\lambda_i} h_{\rho(i)})
\end{aligned}$$

Note that $\lambda_i = \sum_{j=1}^{k} v_j M_{i,j}$, the last equality is derived by:

$$\sum_{i=1}^{\ell} \lambda_i (\alpha_i t + \beta_i) = t \sum_{i=1}^{\ell} \lambda_i \alpha_i + \sum_{i=1}^{\ell} \lambda_i \beta_i = t \cdot 1 + 0 = t.$$