Efficient and Provable Secure Ciphertext-Policy Attribute-Based Encryption Schemes

Luan Ibraimi[1], Qiang Tang[1], Pieter Hartel[1], and Willem Jonker[1,2]

[1] Faculty of EEMCS, University of Twente, the Netherlands
[2] Philips Research, the Netherlands

Abstract. With a Ciphertext-Policy Attribute-Based Encryption (CP-ABE) scheme, a user's private key is associated with a set of attributes and the data is encrypted under an access policy defined by the message sender. A user can decrypt a ciphertext if and only if her attributes satisfy the access policy. In CP-ABE, since the message sender enforces the access policy during the encryption phase, the policy moves with the encrypted data. In this paper, we provide an efficient CP-ABE scheme which can express any access policy represented by a formula involving the *and* ($\wedge$) and *or* ($\vee$) operators. The scheme is secure under Decision Bilinear Diffie-Hellman (DBDH) assumption. Furthermore, we extend the expressiveness of the scheme by including the *of* operator in addition to $\wedge$ and $\vee$. We provide a comparison with some existing CP-ABE schemes and show that our schemes are more efficient.

1 Introduction

Public-key encryption is an asymmetric primitive which uses a pair of keys, where a private key which is kept secret and a public key which is widely distributed. If Alice wants to send a confidential message to Bob, she can encrypt the message with the public key of Bob and only Bob can decrypt the message using his private key. With a Public-Key Infrastructure (PKI), a public key must be obtained from, or at least be certified by the Trusted Third Party (TTP) of the PKI. With an Identity-Based Encryption (IBE) scheme [6,10,19], any string can be used to generate a public key without the involvement of the TTP. IBE thus creates a degree of flexibility that a PKI cannot offer. However, if Alice does not know the identity of her party, but instead she only knows certain attributes of the recipient, then the above solutions will not work.

The solution to this problem is provided by Attribute-Based Encryption (ABE) which identifies a user with a set of attributes [17]. In the seminal paper, Sahai and Waters use biometric measurements as attributes in the following way. A private key, associated with a set of attributes ω, can decrypt a ciphertext encrypted with a public key, associated with a set of attributes ω', only if the sets ω and ω' overlap sufficiently as determined by a threshold value t. We refer to the scheme, proposed by Sahai and Waters, as the SW scheme. A more general policy to decide which attributes are required to decrypt a message is provided by an access tree. For example the access tree $\tau = class1978 \wedge mycollege \vee myteacher$

F. Bao, H. Li, and G. Wang (Eds.): ISPEC 2009, LNCS 5451, pp. 1–12, 2009.

states that all students with the attribute *class*1978 who studied at *mycollege* as well as the teacher possessing the attribute *myteacher* would satisfy the policy.

There are two variants of ABE, namely Key-Policy based ABE (KP-ABE) [12] and Ciphertext-Policy based ABE (CP-ABE)[3,9]. In KP-ABE, the ciphertext is associated with a set of attributes and the private key is associated with the access tree. The message sender does not define the privacy policy and has no control over who has access to the data, except for defining the set of descriptive attributes necessary to decrypt the ciphertext. The trusted authority who generates user's private key defines the combination of attributes for which the private key can be used. In CP-ABE, the idea is reversed: the ciphertext is associated with the access tree and the message sender determines the policy under which the data can be decrypted, while the private key is associated with a set of attributes.

Pirreti *et al.* [16] give a construction and implementation of a modified SW scheme, which, compared to the original scheme, reduces computational overhead during the encryption and the key generation phases. Goyal *et al.* [12] introduce the idea of KP-ABE and propose a new scheme. In their scheme, when a user makes a secret key request, the trusted authority determines which combination of attributes must appear in the ciphertext for the user to decrypt. In essence, this scheme is an extension of the SW scheme, where, instead of using the Shamir [18] secret sharing technique in the private key, the trusted authority uses a more generalized form of secret sharing to enforce a monotonic access tree. Chase [8] constructs a multi-authority ABE scheme, which allows multiple independent authorities to monitor attributes and distribute secret keys. A related work to KP-ABE is a predicate encryption paradigm or searching on encrypted data [1,5,7,14]. Predicate encryption has the advantages of providing ciphertext anonymity by hiding the access structures, however, the system is less expressive compared to schemes which leave the access structures in the clear. Smart [20] gives an access control data scheme which encrypts data to an arbitrary collection of identities using a variant of the Boneh-Franklin IBE scheme.

The first CP-ABE scheme proposed by Bethencourt *et al.* [3] uses threshold secret sharing to enforce the policy during the encryption phase. We refer to this scheme as the BSW scheme. The main drawback of the BSW scheme is that it requires polynomial interpolation to reconstruct the secret, thus many expensive pairing and exponentiation operations are required in the decryption phase. The scheme is secure in the generic group model, which provides evidence to the hardness of the problem, without giving security proof which reduces the problem of breaking the scheme to a well-studied complexity-theoretic problem. The CP-ABE scheme, proposed by Cheung and Newport [9], does not use threshold secret sharing but uses random elements to enforce the policy during the encryption phase. We refer to this scheme as the CN scheme. The CN scheme has two drawbacks. Firstly, it is not sufficiently expressive because it supports only policies with logical conjunction. Secondly, the size of the ciphertext and secret key increases linearly with the total number of attributes in the system. This

makes the CN scheme inefficient. Goyal *et al.* [11] give a "bounded" CP-ABE construction. The disadvantage of their scheme is that the depth of the access trees d under which messages can be encrypted is defined in the setup phase. Thus, the message sender is restricted to use only an access tree which has the depth $d' \leq d$.

Contribution. In this paper we aim at designing efficient CP-ABE schemes, which can be proven based on standard complexity-theoretic assumptions. More specifically, our contribution is twofold:

- We present a new technique for realizing CP-ABE without using Shamir's threshold secret sharing. We first show such a construction which is referred to as as the basic CP-ABE scheme. In this scheme, the message sender defines the privacy policy through an access tree which is n-ary tree represented by $\wedge$ and $\vee$ nodes. Note that, realizing a scheme, which does not use threshold secret sharing, is important for resource constraint devices since calculating polynomial interpolations to construct the secret is computationally expensive. Compared to the CN scheme, our scheme requires fewer computations during the encryption, key generation and decryption phases.
- Next, we extend the basic CP-ABE scheme and provide a second CP-ABE scheme which uses Shamir's threshold secret sharing technique [18]. The access tree is an n-ary tree represented by $\wedge$, $\vee$ and *of* nodes. We compare the efficiency of our scheme with the BSW scheme and show that our scheme requires less computations in the key generation, encryption and decryption phases.

Organization. The rest of this paper is organized as follows. In Section 2 we review concepts of the access structure, secret sharing, CP-ABE, and bilinear pairing. In Section 3 we introduce the basic CP-ABE scheme which is secure under DBDH assumption in the selective-attribute model. In Section 4 we provide an extension to the basic CP-ABE scheme by including the *of* operator in addition to $\wedge$ and $\vee$. In the last section we conclude the paper.

2 Background Knowledge

In this section we introduce the notions related to access structure, secret sharing, the security definition of CP-ABE, and bilinear maps.

2.1 Access Structure

We restate the definition of Access Structure in [2].

Definition 1. (Access Structure). *Let $\{P_1, P_2, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \cdots, P_n\}}$ is monotone if $\forall B, C$: 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 $\{P_1, P_2, \cdots, P_n\}$, i.e., $A \subseteq 2^{\{P_1, P_2, \cdots, P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.*

2.2 Secret Sharing Schemes

In designing our CP-ABE schemes we will make use of two different secret-sharing schemes: unanimous consent control by modular addition scheme and the Shamir secret sharing scheme.

Unanimous Consent Control by Modular Addition Scheme. In a unanimous consent control by modular addition scheme [15], there is a dealer who splits a secret s into t shares in a such way that all shares are required to reconstruct the secret s. To share the secret s, $0 \leq s \leq p-1$ for some integer p, the dealer generates a $t-1$ random numbers s_i such that $1 \leq s_i \leq p-1$, $1 \leq i \leq t-1$ and $s_t = s - \sum_{i=1}^{t-1} s_i \ mod\ p$. The secret s is recovered by: $s = \sum_{i=1}^{t} s_i$. Shares s_i, $1 \leq i \leq t$ are distributed to parties P_i, $1 \leq i \leq t$. For each party P_i, $1 \leq i \leq t$, the shares are random numbers between 0 and $p-1$, thus no party has any information about s except the dealer.

Shamir's Secret Sharing Scheme. In Shamir's secret sharing technique [18] a secret s is divided into n shares in a such way that any subset of t shares, where $t \leq n$, can together reconstruct the secret; no subset smaller than t can reconstruct the secret. The technique is based on polynomial interpolation where a polynomial $y = f(x)$ of degree $t-1$ is uniquely defined by t points (x_i, y_i). The details of the scheme are as follows:

1. Setup. The dealer D wants to distribute the secret $s > 0$ among t users.
   1)D chooses a prime $p > max(s, n)$, and defines $a_0 = s$.
   2)D selects $t-1$ random coefficients $a_1, ..., a_{t-1}, 0 \leq a_j \leq p-1$, and defines the random polynomial over $\mathbb{Z}_p$, $f(x) = \Sigma_{j=0}^{t-1} a_j x^j$.
   3)D computes $s_i = f(i)\ mod\ p$, and sends securely the share s_i to user p_i together with the public index i.
2. Pooling of shares. Any group of t or more users pool their distinct shares $(x, y) = (i, s_i)$ allowing computation of the coefficients a_j of $f(x)$ by Lagrange interpolation, $f(x) = \Sigma_{i=0}^{t-1} l_j(x)$ where $l_j(x) = \Pi_{1 \leq j \leq t, j \neq i} \frac{x - x_j}{x_i - x_j}$. The secret is $f(0) = a_0 = s$.

2.3 Ciphertext-Policy ABE

According to the definition given in [3], a CP-ABE schemes consist of four polynomial time algorithms:

- $\mathsf{Setup}(k)$. The setup algorithm takes as input a security parameter k and outputs the public parameters pk and a master key mk.
- $\mathsf{Keygen}(\omega, mk)$. The algorithm takes as input the master key mk and a set of attributes ω. The algorithm outputs a secret key sk_ω associated with ω.
- $\mathsf{Encrypt}(m, \tau, pk)$. The encryption algorithm takes as input a message m, an access tree τ representing an access structure, and the public key pk. The algorithm will return the ciphertext c_τ such that only users who have the secret key generated from the attributes that satisfy the access tree will be able to decrypt the message.

– $\mathsf{Decrypt}(c_\tau, sk_\omega)$. The decryption algorithm takes as input a ciphertext c_τ, a secret key sk_ω associated with ω, and it outputs a message m or an error symbol $\perp$.

The semantic security against chosen-plaintext attack (CPA) is modeled in the selective-attribute (sAtt) model, where the adversary must provide the challenge access tree he wishes to attack before he receives the public parameters from the challenger. Suppose, that the adversary in the Init phase chooses the challenge access tree $\tau^* = (A \wedge B) \vee C$. In Phase1, the adversary can make secret key requests to Keygen oracle for any attribute set ω with the restriction that attributes $A, B, C \notin \omega$. The selective-attribute (sAtt) model can be considered to be analogous to the selective-ID model [4] used in identity-based encryption schemes, in which the adversary commits ahead of time the ID^* it will attack, and where the adversary can make secret key requests to Keygen oracle for any ID such that $ID \neq ID^*$.

The game is carried out between a challenger and an adversary $\mathcal{A}$, where the challenger simulates the protocol execution and answers queries from $\mathcal{A}$. Specifically, the game is as follows:

1. Init. The adversary chooses the challenge access tree τ^* and gives it to the challenger.
2. Setup. The challenger runs Setup to generate (pk, mk) and gives the public key pk to the adversary $\mathcal{A}$.
3. Phase1. $\mathcal{A}$ makes a secret key request to the Keygen oracle for any attribute set $\omega = \{a_j | a_j \in \Omega\}$, with the restriction that $a_j \notin \tau^*$. The challenger returns $\mathsf{Keygen}(\omega, mk)$.
4. Challenge. $\mathcal{A}$ sends to the challenger two messages m_0, m_1.The challenger picks a random bit $b \in \{0, 1\}$ and returns $c_b = \mathsf{Encrypt}(m_b, \tau^*, pk)$.
5. Phase2. $\mathcal{A}$ can continue querying Keygen with the same restriction as during Phase1.
6. Guess. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$.

Definition 2. *A CP-ABE scheme is said to be secure against an adaptive chosen-plaintext attack (CPA) if any polynomial-time adversary has only a negligible advantage in the IND-sAtt-CPA game, where the advantage is defined to be* $\epsilon = |\Pr[b' = b] - \frac{1}{2}|$.

2.4 Review of Pairing

We briefly review the basis of bilinear pairing. A pairing (or, bilinear map) satisfies the following properties:

1. $\mathbb{G}_0$ and $\mathbb{G}_1$ are two multiplicative groups of prime order p.
2. g is a generator of $\mathbb{G}_0$.
3. $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ is an efficiently-computable bilinear map with the following properties:

- Bilinear: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
- Non-degenerate: $\hat{e}(g, g) \neq 1$.

$\mathbb{G}_0$ is said to be a bilinear group if the group operation in $\mathbb{G}_0$ can be computed efficiently and if there exists a group $\mathbb{G}_1$ and an efficiently-computable bilinear map $\hat{e}$ as defined above.

The Decision Bilinear Diffie-Hellman (DBDH) problem is defined as follows. Given $g, g^a, g^b, g^c \in \mathbb{G}_0$ as input, the adversary must distinguish a valid tuple $\hat{a}(g, g)^{abc} \in \mathbb{G}_1$ from the random element $Z \in \mathbb{G}_1$. An algorithm $\mathcal{A}$ has advantage ϵ in solving the Decision Bilinear Diffie-Hellman (DBDH) problem in $\mathbb{G}_0$ if:

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc}) = 0] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, Z) = 0]| \geq \epsilon.$$

Here the probability is over the random choice of $a, b, c \in \mathbb{Z}_p$, the random choice of $Z \in \mathbb{G}_1$, and the random bits of $\mathcal{A}$ (the adversary is a nondeterministic algorithm).

Definition 3. *We say that the (t, ϵ)-DBDH assumption holds if no t-time algorithm has advantage at least ϵ in solving the DBDH problem in $\mathbb{G}_0$.*

3 The Basic CP-ABE Construction

In this paper, the access tree is a n-ary tree, in which leaves are attributes and inner nodes are $\wedge$ and $\vee$ boolean operators. Intuitively, the access tree is a policy which specifies which combination of attributes can decrypt the ciphertext. Consider the following example where a patient wants to specify access restrictions on his medical data. The patient can enforce the access policy in the encryption phase. Each member from the medical staff who has enough attributes should be able to decrypt the encrypted message. For instance, a patient wants to allow his data to be seen by Doctor A who works at Department A or by Doctor B who works at Department B. Using boolean operators the patient defines the following access policy: $\tau_{Data} = (Doc.A \wedge Dep.A) \vee (Doc.B \wedge Dep.B)$.

To decrypt the ciphertext which is encrypted according to the τ_{Data} access tree, the decryptor must possess a private key, which is associated with the attribute set which satisfies τ_{Data}. To determine whether or not an access tree is satisfied, we interpret each attribute as a logical variable. Possession of the secret key for the corresponding attribute makes the logical variable true. If the decryptor does not possess the attribute, the variable is false. For the policy above there are several different sets of attributes that can satisfy the access tree, such as the secret key associating with the attribute set $\{Doc.A, Dep.A\}$, the secret key associating with the attribute set $\{Doc.B, Dep.B\}$, or the secret key associating with all attributes defined in the access tree.

3.1 Description of the Basic Scheme

The polynomial time algorithms of the basic CP-ABE scheme are defined as follows.

1. $\mathsf{Setup}(k)$: On input of the security parameter k, this algorithm generates the following.
   (a) Generate a bilinear group $\mathbb{G}_0$ of prime order p with a generator g and bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$
   (b) Generate the attribute set $\Omega = \{a_1, a_2, \ldots a_n\}$, for some integer n, and random elements $\alpha, t_1, t_2 \ldots t_n \in \mathbb{Z}_p$.

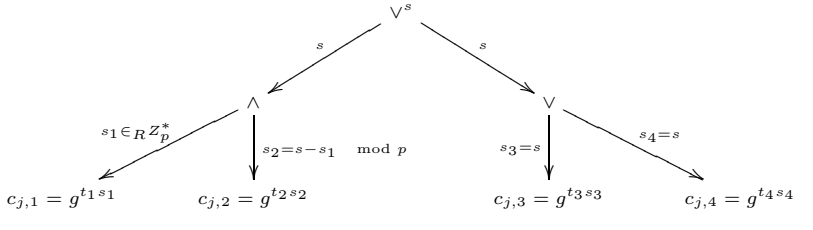
   Let $y = \hat{e}(g,g)^{\alpha}$ and $T_j = g^{t_j}$ $(1 \leq j \leq n)$. The public key is $pk = (\hat{e}, g, y, T_j (1 \leq j \leq n))$ and the master secret key is $mk = (\alpha, t_j (1 \leq j \leq n))$.
2. $\mathsf{Keygen}(\omega, mk)$: The algorithm performs as follows.
   (a) Select a random value $r \in \mathbb{Z}_p$ and compute $d_0 = g^{\alpha - r}$.
   (b) For each attribute a_j in ω, compute $d_j = g^{r t_j^{-1}}$.
   (c) Return the secret key $sk_{\omega} = (d_0, \forall a_j \in \omega : d_j)$
3. $\mathsf{Encrypt}(m, \tau, pk)$: To encrypt a message $m \in \mathbb{G}_1$ the algorithm proceeds as follows:
   (a) First level encryption: Select a random element $s \in \mathbb{Z}_p$ and compute $c_0 = g^s$ and

$$c_1 = m \cdot y^s = m \cdot \hat{e}(g,g)^{\alpha s}$$

   (b) Second level encryption: Set the value of the root node of τ to be s, mark all child nodes as un-assigned, and mark the root node assigned. Recursively, for each un-assigned non-leaf node, do the following:
      – If the symbol is $\wedge$ and its child nodes are marked un-assigned, we use a unanimous consent control by modular addition scheme to assign a value to each child node. To do that, for each child node except the last one, assign a random value s_i where $1 \leq s_i \leq p-1$, and to the last child node assign the value $s_t = s - \sum_{i=1}^{t-1} s_i \mod p$. Mark this node assigned.
      – If the symbol is $\vee$, set the values of each child node to be s. Mark this node assigned.

   Values of the leaves of τ are used to produce ciphertext components.
   (c) For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = T_j^{s_i}$ where i denotes the index of the attribute in the access tree. The index values are uniquely assigned to leave nodes in an ordering manner for a given access structure.
   (d) Return the ciphertext $c_{\tau} = (\tau, c_0, c_1, \forall a_{j,i} \in \tau : c_{j,i})$.

In the following figure, we show an example of assigning secret shares s_i to the access tree $\tau = (T_1 \wedge T_2) \vee (T_3 \vee T_4)$.



4. $\mathsf{Decrypt}(c_\tau, sk_\omega)$: If ω does not satisfy τ, return $\perp$, otherwise the algorithm chooses the smallest set $\omega' \subseteq \omega$ (we assume that this can be computed efficiently by the decryptor) that satisfies τ and performs as follows:

   (a) For every attribute $a_j \in \omega'$, compute

$$\begin{aligned}\prod_{a_j \in \omega'} \hat{e}(c_{j,i}, d_j) &= \prod_{a_j \in \omega'} \hat{e}(T_j^{s_i}, g^{rt_j^{-1}}) \\ &= \prod_{a_j \in \omega'} \hat{e}(g^{t_j s_i}, g^{rt_j^{-1}}) \\ &= \hat{e}(g,g)^{rs}\end{aligned}$$

   (b) Compute

$$\begin{aligned}\hat{e}(c_0, d_0) \cdot \hat{e}(g,g)^{rs} &= \hat{e}(g^s, g^{\alpha - r}) \cdot \hat{e}(g,g)^{rs} \\ &= \hat{e}(g^s, g^\alpha)\end{aligned}$$

   (c) Return m', where

$$\begin{aligned}m' &= \frac{c_1}{\hat{e}(g^s, g^\alpha)} \\ &= \frac{m \cdot \hat{e}(g,g)^{\alpha s}}{\hat{e}(g^s, g^\alpha)} \\ &= m\end{aligned}$$

3.2 Security and Efficiency Analysis

The proposed scheme is proven IND-sAtt-CPA secure under the DBDH assumption. For more details of the proof, the reader can refer to the full version of this paper [13].

The number of calculations in the Encryption algorithm depends on the number of attributes in the access tree τ. The encryption requires $|\tau| + 1$ exponentiations in $\mathbb{G}_0$ and one exponentiation in $\mathbb{G}_1$. The number of calculations in the KeyGen algorithm depends on the number of attributes in the set ω that the user has, namely $|\omega| + 1$ exponentiations in $\mathbb{G}_0$. The number of calculations in the decryption algorithm depends on the number of attributes in the attribute set ω'. The decryption requires $|\omega'| + 1$ pairing operations, $|\omega'|$ multiplications, but no exponentiations in $\mathbb{G}_1$.

In Table 1, we compare our CP-ABE scheme with the CN scheme. We count the number of calculations in the encryption, key generation, and decryption phases. Compared to the CN scheme, our scheme requires fewer computations in the encryption, key generation and decryption phase.

Table 1. The Comparison with the CN Scheme

	Our Scheme			The CN Scheme		
	Exp.($\mathbb{G}_0$)	Exp.($\mathbb{G}_1$)	Pairing	Exp.($\mathbb{G}_0$)	Exp.($\mathbb{G}_1$)	Pairing
Encrypt	$\|\tau\|+1$	1	/	$\|\Omega\|+1$	1	/
Keygen	$\|\omega\|+1$	/	/	$\|\Omega\|+1$	/	/
Decrypt	/	/	$\|\omega'\|+1$	/	/	$\|\Omega\|+1$
	Ω is the set of all attributes defined in the Setup phase τ is the access tree ω is the set of attributes the user has, $\omega \subseteq \Omega$ ω' the set of attributes satisfying the access tree, $\omega' \subseteq \omega$					

4 Extension of the Expressiveness

In the basic scheme, the access tree is a n-ary tree represented by $\wedge$ and $\vee$ nodes, which allows the user who performs encryption to express any privacy policy using boolean formulas. Similar to the BSW scheme, we would like to have an n-ary access tree which supports the *of* operator. The essential idea is to allow the encryptor to define the minimum number of attributes from a given list of attributes that the decryptor has to posses in order to decrypt the message. For instance, to decrypt the ciphertext encrypted under the policy τ =2 *of (class1978, mycollege, myteacher)*, the decryptor must have at least two out of three attributes. We extend the basic CP-ABE scheme to support the *of* operator as follows:

1. Setup and KeyGen are the same as in basic CP-ABE scheme.
2. Encrypt(m, τ, pk) : To encrypt a message $m \in \mathbb{G}_1$ the algorithm proceeds as follows:

   (a) First level encryption: Select a random element $s \in \mathbb{Z}_p$ and compute $c_0 = g^s$ and

   $$\begin{aligned} c_1 &= m \cdot y^s \\ &= m \cdot \hat{e}(g,g)^{\alpha s} \end{aligned}$$

   (b) Second level encryption: Set the value of the root node to be s, mark all child nodes as un-assigned, and mark the root node assigned. Recursively, for each un-assigned non-leaf node, do the following:

   - If the symbol is *of* (threshold operator), and its child nodes are marked un-assigned, the secret s is divided using (t, n) Shamir's secret sharing technique where $t \neq n$, and n is the total number of child nodes and t is the number of child nodes necessary to reconstruct the secret. To each child node a share secret $s_i = f(i)$ is assigned. Mark this node assigned.
   - If the symbol is $\wedge$, and its child nodes are marked un-assigned, the secret s is divided using (t, n) Shamir's secret sharing technique where $t = n$, and n is the number of the child nodes. To each child node a share secret $s_i = f(i)$ is assigned. Mark this node assigned.

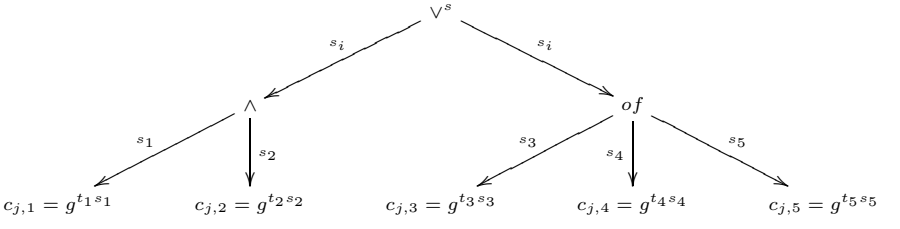
– If the symbol is $\vee$, and its child nodes are marked un-assigned, the secret s is divided using (t, n) Shamir's secret sharing technique where $t = 1$ and n is the number of the child nodes. To each child node a share secret $s_i = f(i)$ is assigned. Mark this node assigned.

Values of the leaves of τ are used to produce ciphertext components.

(c) For each leaf attribute $a_{j,i} \in \tau$, compute $c_{j,i} = T_j^{s_i}$, where i denotes the index of the attribute in the access tree.

(d) Return the ciphertext: $c_\tau = (\tau, c_0, c_1, \forall a_{j,i} \in \tau : c_{j,i})$.

In the following figure, we show an example of assigning secret shares s_i to the access tree $\tau = (T_1 \wedge T_2) \vee 2\ of\ (T_3, T_4, T_5)$.



3. $\mathsf{Decrypt}(c_\tau, sk_\omega)$: If ω does not satisfy τ, return $\perp$, otherwise the algorithm chooses the smallest set $\omega' \subseteq \omega$ that satisfies τ and performs as follows:

(a) For every attribute $a_j \in \omega'$, compute

$$\begin{aligned}\prod_{a_j \in \omega'} \hat{e}(c_{j,i}, d_j)^{l_i(0)} &= \hat{e}(T_j^{s_i}, g^{rt_j^{-1}})^{l_i(0)} \\ &= \prod_{a_j \in \omega'} \hat{e}(g^{t_j s_i}, g^{rt_j^{-1}})^{l_i(0)} \\ &= \prod_{a_j \in \omega'} \hat{e}(g, g)^{r s_i l_i(0)} \\ &= \hat{e}(g, g)^{rs}\end{aligned}$$

$l_i(0)$ is a Lagrange coefficient and can be computed by everyone who knows the index of the attribute in the access tree.

(b) Compute

$$\begin{aligned}\hat{e}(c_0, d_0) \cdot \hat{e}(g, g)^{rs} &= \hat{e}(g^s, g^{\alpha - r}) \cdot \hat{e}(g, g)^{rs} \\ &= \hat{e}(g^s, g^\alpha)\end{aligned}$$

(c) Return m', where

$$m' = \frac{c_1}{\hat{e}(g^s, g^\alpha)} = \frac{m \cdot \hat{e}(g, g)^{\alpha s}}{\hat{e}(g^s, g^\alpha)} = m$$

Table 2. The Comparison with the BSW scheme

	Our Scheme			The BSW Scheme		
	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_1$)	Pairing	Exp.($\mathbb{G}$)	Exp.($\mathbb{G}_1$)	Pairing
Encrypt	$\|\tau\|+1$	1	/	$2\|\tau\|+1$	1	/
KeyGen	$\|\omega\|+1$	/	/	$2\|\omega\|+1$	/	/
Decrypt	/	$\|\omega'\|$	$\|\omega'\|+1$	/	$\|\omega'\|$	$2\|\omega'\|$
(Note:)	Ω is the set of all attributes defined in the Setup phase τ is the access tree ω is the set of attributes the user has, $\omega \subseteq \Omega$ ω' the set of attributes satisfying the access tree, $\omega' \subseteq \omega$					

The proposed scheme is proven IND-sAtt-CPA secure under the DBDH assumption as shown in the full version of this paper [13]. In Table 2, we give a comparison of the efficiency of our extended CP-ABE scheme with the BSW scheme. Compared to the BSW scheme, our scheme requires fewer computations in the encryption, key generation and decryption phases.

5 Conclusion and Future Work

We have shown how to improve the efficiency of a CP-ABE scheme. Firstly, we presented a new technique to construct a CP-ABE scheme which does not use threshold secret sharing. The encryptor specifies the policy in the encryption phase using an n-ary tree which consists from $\vee$ and $\wedge$ nodes. Secondly, we presented a modified scheme which is more expressive compared to the basic scheme. In the modified scheme, the policy can be expressed as an n-ary access tree which consists of $\vee$, $\wedge$ and *of* nodes. We have shown that these schemes require less computations than some other similar schemes.

Acknowledgments

We thank Asim Muhammad and Peter Van Liesdonk for their suggestions and comments.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 205–222. Springer, Heidelberg (2005)
2. Beimel, A.: Secure Schemes for Secret Sharing and Key Distribution, PhD. Thesis, Department of Computer Science,Technion (1996)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-Policy Attribute-Based Encryption. In: Proceedings of the 2007 IEEE Symposium on Security and Privacy, pp. 321–334. IEEE Computer Society, Los Alamitos (2007)
4. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)

5. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
6. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
7. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (Without random oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
8. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
9. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 456–465. ACM, New York (2007)
10. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001)
11. Goyal, V., Jain, A., Pandey, O., Sahai, A.: Bounded ciphertext policy attribute based encryption. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part II. LNCS, vol. 5126, pp. 579–591. Springer, Heidelberg (2008)
12. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98. ACM, New York (2006)
13. Ibraimi, L., Tang, Q., Hartel, P., Jonker, W.: Efficient and provable secure ciphertext-policy attribute-based encryption schemes. Technical Report TR-CTIT-08-75, CTIT, University of Twente (2008)
14. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
15. Menezes, A., Oorschot, P.V., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
16. Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: Secure attribute-based systems. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 99–112 (2006)
17. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
18. Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)
19. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
20. Smart, N.P.: Access control using pairing based cryptography. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 111–121. Springer, Heidelberg (2003)