A CP-ABE Scheme with Hidden Policy and its Application in Cloud Computing

Abstract: With flexible and scalable features for fine-grained access control, Ciphertext Policy Attribute-based Encryption (CP-ABE) is widely used as a kind of data protection mechanism in cloud computing. However, the access policy of CP-ABE scheme may contain sensitive information which causes privacy revelation of the data provider or receiver. Some papers proposed hidden policy CP-ABE schemes, which are based on And-gate access structure whose expressive ability of access policy is limited. CP-ABE with the tree-based access structure has stronger expressive ability and more flexible access control capability. Therefore, it has broad application prospects compared to other mechanisms. This paper proposes a tree-based access structure CP-ABE scheme with hidden policy (CP-ABE-HP), and also proves that the scheme has Chosen-plaintext Attack (CPA) security. CP-ABE-HP can both protect the policy and has flexible access control capability. Then, considering the characteristics of cloud computing environment, the paper constructs a new self-contained data protection mechanism based on CP-ABE-HP, which can provide reliable and flexible security control to the data in cloud.

Keywords: Ciphertext Policy Attribute-based Encryption; Access Control; Hidden Policy; Self-contained Data Protection; Cloud Computing.

Reference

Biographical notes:

1 Introduction

With the development of cloud computing, the cloud storage is becoming a popular way of data storage for enterprises or individuals. In the new way of data storage, data is out of the user's control domain and the cloud storage service provider may be unreliable. Therefore, the protection mechanism of outsourced data attracts much more attentions, and the cloud storage environment requires the data with the self-protection capability.

Data encryption, currently the primary means to protect data, cannot satisfy data protection requirements in various online applications which own a large amount of users, due to its complex key management mechanism and poor scalability. Hence, a new data oriented protection mechanism is urgently needed, in which data can protect its confidentiality and integrity all by itself rather than depending on the cloud storage servers. We call this kind of new data protection the data-centric self-contained protection. Also we believe that this kind of data protection can be achieved by integrating encryption with access control.

In recent years, for the situation of uncertain decrypting party, Attribute-based Encryption (ABE) (Sahai and Waters, 2005) mechanism was developed based on Identity-based encryption mechanism. Without knowing the specific decrypting parties, the data provider encrypts data according to an access structure consisting of a series of attribute descriptions, and the data receiver can decrypt the ciphertext only if he/she satisfies the

attribute descriptions in the access structure. With data decryption depending on user attributes, ABE can solve the security problems of outsourced data effectively. At present, ABE can be divided into two types: Key Policy Attribute-based Encryption (KP-ABE) and Ciphertext Policy Attribute-based Encryption (CP-ABE). In KP-ABE (Goyal et al., 2006), the ciphertext is associated with a set of attributes and the secret key is associated with the access structure. On the contrary, in CP-ABE (Bethencourt et al., 2007), the ciphertext is associated with the access structure and the secret key is associated with a set of attributes. Consequently, CP-ABE scheme is more suitable for data-centric self-contained protection in cloud storage environment. And CP-ABE scheme currently has three kinds of access structures: And-gate access structure, tree-based access structure and Linear Secret Share Scheme (LSSS) matrix access structure. The tree-based access structure can express much more complex access policy with its hierarchy. Thus, it has a more flexible access control capability for data encryption. Therefore, CP-ABE has gained much more attentions in cloud computing.

In the originally proposed CP-ABE schemes, the access structure is embedded in the ciphertext and whoever obtains the ciphertext can see the content of the access structure. However, this full exposure of data's access policy will disclose sensitive information of the decryption or encryption party. For example, in commercial environments, the access structure may contain trade secrets; an access structure for secret job information released by a company on public platform may disclose the company's development direction and strategy. In military applications, the access structures themselves may contain military secrets, such as sensitive information like organizational structure, the core combat troops, staff ratio, and firepower configuration of a group army. When utilizing ABE to protect shared data on the Internet, the access policy may also disclose the receivers' privacy information. Meanwhile, in order to avoid the attack by malicious users and policy-based inference for important information, the policy should be hidden.

The anonymous ABE schemes (Frikken et al., 2006; Yu et al., 2008; Nishide et al., 2008) give a good solution to these problems. These schemes protect users' privacy information by establishing security protocols and using encryption to protect access policy against unauthorized access in the security protocols. Recently, the researchers also proposed a series of CP-ABE schemes(Li et al., 2012; Lai et al., 2011; Doshi and Jinwala, 2012) with hidden policy. But these schemes are based on simple And-gate access structure and their policy expressive ability is limited. To the best of our knowledge, the tree-based access structure CP-ABE scheme with hidden policy has not been proposed at present.

Meanwhile, in the current applications based on CP-ABE, the user of encryption party or decryption party should execute a series of operations, such as authentication, private key acquisition, encryption and decryption. These operations add burden to users and also affect the efficiency of applications. Hence, in the cloud computing, designing a general data protection mechanism which supports CP-ABE and achieves the basic operations of protected data is essential. The mechanism can achieve the self-contained data protection and provide transparent data protection services to the users.

This paper firstly proposes a tree-based access structure CP-ABE scheme with hidden policy (CP-ABE-HP) which is also proved to have the Chosen-plaintext Attack (CPA) security under the Decisional Bilinear Diffie-Hellman (DBDH) assumption in the standard model. Inspired by the tree-based access structure in ITHJ09 scheme (Ibraimi et al., 2009) and the policy hiding method in And-gate access structure CP-ABE (Li et al., 2012), we proposes a more efficient and stronger expressive CP-ABE scheme with hidden policy. Our scheme also uses subgroup element's orthogonal property in composite order bilinear

groups and introduces some random elements into the policy key component. Then, the paper proposes a self-contained data protection mechanism and builds an user-friendly system called Easy Share which implements the mechanism.

The remaining sections are organized as follows. In Section 2, we introduce the related work. In Section 3, we review some preliminary concepts. We propose the CP-ABE-HP scheme with analysis and give the security proof of our scheme in Section 4. In Section 5 we introduce the self-contained data protection mechanism and its implementation. Finally, we conclude the paper in Section 6.

2 Related work

BSW07 scheme (Bethencourt et al., 2007) uses tree-based access structure to express access policy, which supports *AND*, *OR* and *of* threshold operators. Its ciphertext length, time of encryption and decryption, and the number of attributes in the access structure are linear correlation. But the security proof is based on common group model, rather than the standard numerical theoretical assumptions. Cheung and Newport primarily constructes the CPA security CP-ABE mechanism (CN07)(Cheung and Newport, 2007) based on the DBDH assumption. However, its access structure only supports *AND*, *OR* operation with weak expressive ability. Also, the ciphertext and key length are linear with the number of system attributes and scheme efficiency is lower.

The Bounded Ciphertext Policy Attribute-based Encryption (BCP-ABE) (Goyal et al., 2008) with tree-based access structure supports *AND*, *OR* and *of* threshold operators, but the height of the tree and the number of child non-leaf nodes are limited. The ITHJ09 scheme (Ibraimi et al., 2009) is based on DBDH assumption using Shamir secret sharing technology (Shamir, 1979) to support *AND*, *OR* and *of* threshold operators. Its access structure is an n-ary tree and its key generation and decryption or encryption cost is lower than BSW07 scheme. Waters firstly used the Linear Secret Sharing Scheme matrix to express access policy (Waters, 2011).

Kapadia et al gives a scheme with hidden certificate and hidden policy based on PEAPOD system (Kapadia et al., 2007). The scheme introduces an online semi-trusted server, but cannot prevent collusion attacks. To prevent users' collusion attacks, two kinds of anonymous CP-ABE mechanisms (Yu et al., 2008, 2010) are constructed to use in CDN network and multicast user groups. However, these anonymous mechanisms uses strong security assumption, so the security level is lower. Nishide et al firstly proposes anonymous CP-ABE with hidden policy based on DBDH assumption and D-Linear assumption (Nishide et al., 2008). But the mechanism only supports AND-gate access structure. In papers (Lai et al., 2011; Balu and Kuppusamy, 2010a; Yu, 2010; Balu and Kuppusamy, 2010b), the authors have proposed different ways to deal with policy hiding issues. Lai et al uses inner product Predicate Encryption technology to achieve the hidden policy CP-ABE scheme in fully security model(Lai et al., 2011). After that, they also proposes a partial hidden policy scheme (Lai et al., 2012) based on LSSS matrix access structure and pointes out that the structure is more flexible than other scheme (Nishide et al., 2008; Lai et al., 2011; Li et al., 2009). Among these schemes, the privacy-aware ABE proposed by Jin et al aimes at the prevention of users' collusion attacks. And their main idea is binding the user's ID to detect whether a user shares their property keys or not. Balu et al calculates dual key for each attribute element to achieve anonymous policy or privacy preserving without supporting of threshold operator (Balu and Kuppusamy, 2010a,b). With And-gate access structure

supporting negative attribute and wildcard, a hidden policy scheme (Doshi and Jinwala, 2012) focuses on the constant size of ciphertext and key. Xiaohui et al also proposes a hidden policy scheme used And-gate access structure with provably security under the standard model, and it is based on Waters' scheme (Waters, 2005).

Considering the security and expressive ability of access policy, only the W08 and ITHJ09 support the *AND*, *OR* and threshold operations under the standard numerical theoretical assumptions, and the computation cost of encryption and decryption of ITHJ09 is lower than W08's. Meanwhile, in terms of hidden policy, all existing CP-ABE schemes are based on And-gate access structure. Though their efficiency has been improved, the expressive ability of policy is limited. In the background of cloud storage applications, CP-ABE scheme with flexible policy expression ability will have broad application prospects. Therefore, the paper focuses on the research of tree-based access structure CP-ABE scheme with hidden policy.

3 Preliminaries

3.1 Composite order bilinear groups

Composite order bilinear groups is first introduced by (Boneh et al., 2005). The order of bilinear groups we used is the product of two distinct primes. Let p, r be distinct primes, G and G_T be cyclic groups of order $N = pr$. And $e : G \times G \to G_T$ is a map satisfied the following conditions:

- Bilinear: $\forall g, h \in G, a, b \in Z_N, e(g^a, h^b) = e(g, h)^{ab}$.
- Non-degenerate: $\exists g \in G$ such that $e(g, h)$ has order N in G_T.

We use G_p and G_r to denote the subgroups of G with order p and r respectively. Note also that if $h_p \in G_p$ and $h_r \in G_r$ then $e(h_r, h_p) = 1$.

3.2 The decisional bilinear Diffie-Hellman assumption

In this paper, we use DBDH assumption as the complexity assumption. Let $e : G \times G \to G_T$ be an efficiently computable bilinear map and g is the generator of G. Choose random numbers $a, b, c, z \in Z_p$. The DBDH assumption is that no probabilistic polynomial-time algorithm β can distinguish the tuple $(g, g^a, g^b, g^c, e(g, g)^{abc})$ from the tuple $(g, g^a, g^b, g^c, e(g, g)^z)$ with more than a negligible advantage.

3.3 Access structure

Definition 1: Access Structure (Beimel, 1996): Let $\{p_1, p_2, ..., p_n\}$ be a set of parties. A collection $A \in 2^{\{p_1, p_2, ..., p_n\}}$ is monotone if $\forall B, C$: if $B \in A \wedge B \subseteq C$ then $C \in A$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) A of non-empty subsets $\{p_1, p_2, ..., p_n\}$, i.e., $A \in 2^{\{p_1, p_2, ..., p_n\}} \setminus \{\emptyset\}$. The sets in A are called the authorized sets, and the sets not in A are called the unauthorized sets.

In CP-ABE-HP mechanism, we use attributes instead of the p_i and the access structure A will contain the set of authorized attributes.

3.4 CP-ABE

The ciphertext-policy attribute based encryption(CP-ABE) scheme consists of four fundamental algorithms (Bethencourt et al., 2007): Setup, Encrypt, Key Generation, and Decrypt.

- Setup (k). The setup algorithm takes no input other than the security parameter k. It outputs the public parameters PK and a master key MK.
- Key-Generation (MK, S). The key generation algorithm takes the master key MK and a set of attributes S that describe the key as input. It outputs a private key SK.
- Encrypt (PK, M, A). The encryption algorithm takes the public parameters PK, a message M and an access structure A over the universe of attributes as input. The algorithm will encrypt M and produce a ciphertext C_T , so that only a user that possesses a set of attributes that satisfies the access structure will be able to decrypt the message. We will assume that the ciphertext implicitly contains A.
- Decrypt (PK, CT, SK). The decryption algorithm takes the public parameters PK, a ciphertext C_T which contains an access policy A, and a private key SK as input. If the attributes set satisfies the access structure A, the algorithm will decrypt the ciphertext and return a message M, otherwise return the error symbol $\perp$.

3.5 Security model

The CPA semantic security model of CP-ABE-HP will be based on the IND-sAtt-CPA game(Ibraimi et al., 2009), which is a simulation between a challenger and an adversary A. In the game, the challenger simulates an execution environment of algorithms to answer the adversary's query request. The specific game process is as follows:

- Init Phase. The adversary chooses a challenge access tree and gives it to the challenger.
- Setup Phase. The challenger runs Setup algorithm to generate (PK, MK) and gives the public key PK to adversary A.
- Phase 1. Adversary A makes a secret key request to the key generation oracle for any attribute sets. The challenger runs Key-Generation (MK, S) algorithm to generate a private key.
- Challenge Phase. Adversary A sends to the challenger two equal length messages m_0, m_1. The challenger picks a random bit $b \in 0, 1$ and returns c_b = Encrypt(m_b, τ^*, PK).
- Phase 2. Adversary A can continue querying key generation oracle with the same restriction as in Phase 1.
- Guess Phase. Adversary A outputs a guess $b' \in 0, 1$.

Definition 2: if the attack advantage of the adversary is ignored in the IND-sAtt-CPA game in any polynomial time, the CP-ABE-HP scheme can be at CPA security. And the advantage is $\varepsilon = |Pr - 1/2|$.

4 CP-ABE-HP

4.1 CP-ABE-HP scheme

The specific CP-ABE-HP scheme is as follows:

1) **Setup(k)**: the algorithm takes security parameter k as input and generates the following parameters.

   a) Generate the bilinear groups G and a bilinear map $e : G \times G \rightarrow GT$, and G and G_T are the cyclic groups of order $N = pr$, where the p and r are distinct primes. Let G_p and G_r be the subgroup of the G with order p and r respectively. Also g_p and g_r are the generator of G_p and G_r respectively.

   b) Generate the attribute set $U = \{a_1, a_2, , a_n\}$, random element $, t_1, t_2, , t_n \in Z_p^*$ and $R_0, R_1, R_2, , R_n \in G_r$. Calculate the public key as follows:

$$x = g_p \cdot R_0 \tag{1}$$

$$y = e(g_p, g_p)^{\alpha} \tag{2}$$

$$T_j = g_p^{t_j} \cdot R_j (1 \leq j \leq n) \tag{3}$$

   So, the public key is $pk = \{e, x, y, T_j (1 \leq j \leq n)\}$, and the master key is $mk = \{\alpha, t_j (1 \leq j \leq n)\}$.

2) **Key-Generation(w, mk)**:the algorithm takes w and mk as the input, where w is the attribute set submited by the user and mk is the master key. The detail algorithm is as follows.

   a) Choose a random element $r \in Z_p^*$, calculate:

$$d_0 = g_p^{\alpha - r} \tag{4}$$

   b) For every attribute $a_j \in w$, calculate:

$$d_j = g_p^{r t_j^{-1}} \tag{5}$$

   Return the secret key $sk_w = \{d_0, \forall a_j \in w : d_j\}$.

3) **Encrypt(m, τ, pk)**: the algorithm encrypts a message $m \in G_T$ as follows, where m is the message, τ is the access policy tree and pk is the public key of the system.

   a) Select a random element $s \in Z_p^*, R_0' \in G_r$, calculate:

$$c_0 = x^s \cdot R_0' \tag{6}$$

$$c_1 = m \cdot y^s = m \cdot e(g_p, g_p)^{\alpha s} \tag{7}$$

   b) Assign the secret s in the tree-based access policy, set the value of the root node of τ to be s. Make all child nodes as un-assigned and the root node as assigned. Recursively, for each un-assigned non-leaf node, do as follows:

- If the node operator is *of* (threshold operator) and its child nodes are un-assigned, the secret s is divided by (t, n)-Shamir Secret Sharing, where n is the number of all child nodes and t is number of child nodes for recover secret s. For each child node, its sharing secret value is $s_i = f(i)$ and mark this node as assigned.
- If the node operator is $\wedge$ and its child nodes are un-assigned, ibid, using (t, n)-Shamir Secret Sharing to share the secret s, where $t = n$. For each child node, its sharing secret value is $s_i = f(i)$ and mark this node as assigned.
- If the node operator is $\vee$ and its child nodes are un-assigned, ibid, using (t, n)-Shamir Secret Sharing to share the secret s, where $t = 1$. For each child node, its sharing secret value is $s_i = f(i)$ and make this node as assigned.

Note that i denotes the position index of the leaf node and the value of each leaf node is used to generate the ciphertext component. The function $f(x)$ is a random polynomial over Z_p^*, and defined as $f(x) = \sum_{j=0}^{t-1} b_j x^j$,where b_j is a random coefficient and t is the number of child nodes.

c) For each leaf node, calculate as follows:

$$\forall a_{j,i} \in \tau, c_{j,i} = T_j^{s_i} \cdot R'_j \tag{8}$$

Where i denotes the index of leaf node in the tree, and R'_j is a random element in G_r group.

4) **Decrypt(c_τ, sk_w)**: the algorithm is described as follows:

$$m' = \frac{c_1}{e(c_0, d_0) \cdot \prod_{a_j \in w} e(c_{j,i}, d_j)^{l_i(0)}} \tag{9}$$

Where $l_i(0)$ is the Lagrange coefficient, can be calculated by the attribute index i, which can be found in the ciphertext components of the attributes, namely, $[i, c_{j,i}]$. And the input parameters c_τ, sk_w denote the ciphertext, the users' private key respectively.

4.2 Analysis

Correctness Proof

We give the correctness proof as follows:

$$\begin{aligned} m' &= \frac{c_1}{e(c_0, d_0) \cdot \prod_{a_j \in w} e(c_{j,i}, d_j)^{l_i(0)}} \\ &= \frac{m \cdot e(g_p, g_p)^{\alpha s}}{e(g_p^s, g^{\alpha - r}) \cdot e(R_0^s R'_0, g_p^{\alpha - r})} \\ &\cdot \frac{1}{\prod_{a_j \in w} (e(g_p^{t_j s_i}, g_p^{r t_j^{-1}})^{l_i(0)} \cdot e(R_j^{s_i} R'_j, g_p^{r t_j^{-1}})^{l_i(0)})} \\ &= \frac{m \cdot e(g_p, g_p)^{\alpha s}}{e(g_p^s, g^{\alpha - r}) \cdot e(g_p, g_p)^{\sum r s_i l_i(0)}} \\ &= \frac{m \cdot e(g_p, g_p)^{\alpha s}}{e(g_p^s, g^{\alpha - r}) \cdot e(g_p, g_p)^{rs}} \\ &= m \end{aligned} \tag{10}$$

Table 1 Comparison of our scheme with other schemes in computing cost

Scheme	Access Structure	Hidden Policy	Encrypt	Decrypt
CN07	And-gate	N	$(n+1)G+2G_t$	$(n+1)C_e+(n+1)G_t$
Emura09	And-gate	N	$(n+1)G+2G_t$	$2C_e+2G_t$
Xiao12	And-gate	Y	$(n+3)G+2G_t$	$2C_e+2G_t$
BSW07	Tree	N	$(2\|A_c\|+1)G+2G_t$	$2\|A_u\|C_e+(2\|S\|+2)G_t$
ITHJ09	Tree	N	$(\|A_c\|+1)G+2G_t$	$(\|w\|+1)C_e+(\|w\|+1)G_t$
CP-ABE-HP	Tree	Y	$2(\|A_c\|+1)G+2G_t$	$(\|w\|+1)C_e+(\|w\|+1)G_t$

Note: G and G_t represent the computing on G and G_t groups respectively. $|w|$ is the number of user's attributes. Ce denotes the bilinear map computing. $|A_c|$ stands for the attribute number in the access structure. $|A_u|$ is the leaf node number in the access structure. $|S|$ indicates the number of user's attribute associated with the private key.

In previous schemes (Bethencourt et al., 2007; Li et al., 2012; Lai et al., 2011; Doshi and Jinwala, 2012; Ibraimi et al., 2009; Cheung and Newport, 2007), the access policy is appended to the ciphertext. Because the access policy is public, it's useless to protect the ciphertext components of the attributes that are associated with the tree-based access policy. However, when the policy is hidden, the only way to find some information about the access policy is to attack the ciphertext components of the attributes. Hence, it is necessary to protect these parts of the ciphertext. In our scheme, we use the property of composite order bilinear groups to achieve the goal of anonymous attributes of receivers. The most important part is that the random element is introduced into ciphertext of c_0 and $c_{j,i}$. In encryption phase, c_0 and $c_{j,i}$ multiplies by the random elements R_0' and R' of G_r respectively, as shown in equation (6) and (8). Meanwhile, it does not affect the decryption result in the decryption phase, as shown in equation (10). So, it can effectively prevent some malicious attacker from testing the access policy by a possible access structure w, guessing the access structures, and getting the anonymous information of receivers.

The performance analysis of the computing efficiency of CP-ABE-HP is shown in Table 1. CN07, Emura09 and Xiao12 are the CP-ABE schemes based on And-gate access structure. Emura09 and Xiao12 scheme have the constant size of the ciphertext and the private key, and Xiao12 scheme realizes a hidden policy scheme. Compared to Emura09 scheme, the Xiao12 scheme has 2 additional computing costs in G group during encryption phase to achieve hidden policy, which is necessary for goal of policy hiding. BSW07, ITHJ09 and our scheme all use tree-based access structure, and our scheme increases one and ($|A_c|$+1) computing cost on G group compared with BSW07 and ITHJ09 respectively, however the computing is only the non-exponentiation on G group. Compared with And-gate hidden access policy in Xiao12 scheme, the computing consumption of our scheme is more than Xiao12 scheme during the decryption phase. The reason is that Xiao12 scheme has the constant size of ciphertext and private key. But during the encryption phase, the computing cost of our scheme is lower than Xiao12 scheme.

4.3 Security proof

In this section, we give the security proof of CP-ABE-HP scheme. Firstly, we suppose that the IND-sAtt-CPA game can be won by an adversary A with a non-negligible advantage ε. We will build a simulator β, which has the ability to solve the DBDH assumption problem with advantage $\varepsilon/2$ from the attack ability of adversary A. The simulator firstly sets the bilinear group G of order $N=pr$ and the bilinear map $e: G\times G\to G_T$, where p and r are the distinct primes and G and G_T are cyclic groups. Let G_p and G_r be the subgroup of G with order p and r and generator g_p and g_r respectively. The challenger selects $u=_R \{0,1\}$ and sets Z_u as follows:if $u=0, Z_u=e(g_p,g_p)^{\theta}$;if $u=1, Z_u=e(g_p,g_p)^{abc}$.And

then the challenger sends a DBDH challenge $(g_p, A, B, C, Z_u) = (g_p, g_p^a, g_p^b, g_p^c, Z_u)$ to the simulator.

In the attack game, the simulator plays the challenger role of adversary and we refer to it as the challenger in the following IND-sAtt-CPA game:

- Init Phase. The adversary chooses a challenge access τ^* and sends it to the challenger.
- Setup Phase. The challenger selects a random element $x' \in Z_p$, sets $= ab + x'$ and calculates $y = e(g_p, g_p)^{ab} e(g_p, g_p)^{x'}$. For all attributes a_j, if $a_j \in \tau^*, T_j = g_p^{t_j} \cdot R_j$; if $a_j \notin \tau^*, T_j = g_p^{b/t_j} \cdot R_j$, where $t_j \in_R Z_p^*, R_j, R_0 \in_R G_r, (1 \le j \le n)$. After setting the parameters, the challenger sends the adversary A the following public key $pk = \{x = g_p \cdot R_0, y, T_j, (1 \le j \le n)\}$.
- Phase 1. The adversary sends a user private key query request to the challenger by any attributes set $w_j = \{a_j | a_j \in \Omega\}, (a_j \notin \tau^*)$. For each query request of the adversary, the challenger selects random element $r' \in_R Z_p$ and sets $r = ab + r'b$, so $d_0 = g_p^{\alpha - (ab + r'b)} = g_p^{x' - r'b} = g_p^{x'} (g_p^b)^{-r'}$. As the restriction $a_j \notin \tau^*$ in the attributes set of private key request from the adversary, $d_j = g_p^{rt_j/b} = (g_p^a)^{t_j} g_p^{r't_j}$. And the challenger sends the adversary the user private key: $sk_w = \{d_0, \forall a_j \in w_j : d_j\}$.
- Challenge Phase. The adversary submits two plaintext messages m_0, m_1 to the challenger. And the challenger selects a random plaintext message m_b from the two messages, where $b \in_R 0, 1$. Encrypt the message, $c_0 = g_p^c \cdot R_0^c \cdot R_0', c_1 = m_b e(g_p, g_p)^{abc} e(g_p^c, g_p^{x'})$. Then set the root node value of challenge tree τ^* to be g_p^c, and initialize all child nodes as un-assigned and mark the root node as assigned. Recursively, for each un-assigned non-leaf, if the node's child nodes are un-assigned, the challenger select a polynomial $f(i)$, i donating the attribute index of challenge tree and $f(0) = c$. For each child node, the challenger assigns a value $g^{f(i)}$, and marks this node as assigned. The polynomial $f(i)$ is set with the following rule:
  - If the node symbol is *of* (threshold operator), set the polynomial $f(i)$ of degree $t - 1$, where t denotes the number of nodes to recover the secret.
  - If the node symbol is $\wedge$, set the polynomial $f(i)$ of degree $n - 1$, where n denotes the number of all leaf nodes.
  - If the node symbol is $\vee$, set the polynomial $f(i)$ of degree 0, so the $f(i)$ is constant number and assign it to each of its child node.
- Phase 2. The adversary continues to send the secret key requests to the challenger with the same restriction as in Phase 1.
- Guess Phase. The adversary outputs a guess $b' \in \{0, 1\}$.

Analysis: If $b' = b$, the challenger can guess that $u = 0, Z_u = e(g_p, g_p)^{abc}$. As $Z_u = e(g_p, g_p)^{abc}$ is a reasonable simulation of the simulator, the ciphertext is a valid ciphertext in the system. Hence, with the help of adversary, the challenger solves the DBDH assumption problem with the advantage $Pr[b' = b | Z_u = e(g_p, g_p)^{abc}] = 1/2 + \varepsilon$. Otherwise, the challenger guesses that $u = 0, Z_u = e(g_p, g_p)$. Right now the value of $Z_u = e(g_p, g_p)$ is a random ciphertext relative to the adversary. And the adversary cannot get any information about the plaintext message m_b. So, the challenger solves the DBDH assumption problem

with the advantage $Pr[b' \neq b|Z_u = e(g_p, g_p)^{abc}] = 1/2$. Conclusion as a result, for any guesses, the challenger solves the DBDH assumption problem with the following advantage:

$$\frac{1}{2}Pr[u' = u|u = 0] + \frac{1}{2}Pr[u' = u|u = 1] - \frac{1}{2} = \frac{\varepsilon}{2} \tag{11}$$

In summary, the elements, like R_j, R'_0, from the G_r group are random and one-time elements. Compared to the previous schemes [3,7-11], the random elements do not lead to new security problems. Hence, we focus on the same security model as before. If the adversary has the above advantage ε to win the IND-sAtt-CPA game, the challenger will solve the DBDH assumption problem with advantage $\varepsilon/2$ by the help of the adversary's advantage. However, there are no effective polynomial algorithms which can solve the DBDH assumption problem with non-negligible advantage according to the DBDH assumption. Hence, the adversary also cannot win the IND-sAtt-CPA game with the above advantage ε, namely, the adversary having no advantage to break through CP-ABE-HP system.

5 Self-contained data protection mechanism and its implementation

The traditional architecture based on the CP-ABE schemes have tree parties: an encryption party, a decryption party and a Private Key Generator(PKG). The PKG initializes the system and distributes the public parameters. Then the encryption party encrypts the data with his access policy. And the decryption party requests the private key from the PKG and decrypts the ciphertext. However, in cloud computing environment, with the increasing amount of data and users, the encryption and decryption operations will be a heavy burden to the users. Also, it will affects the efficiency of the cloud computing application.

Therefore, to achieve the self-contained data protection mechanism based on CP-ABE-HP and reduce the users' operation burden, we import the concept of container from the web server and J2EE architectures. Basing on the analysis of the characteristics and operating requirements of the CP-ABE-HP encrypted data, we propose the cloud computing oriented Data Security Container (DSC). We developed the system named Easy Share which realized the DSC.

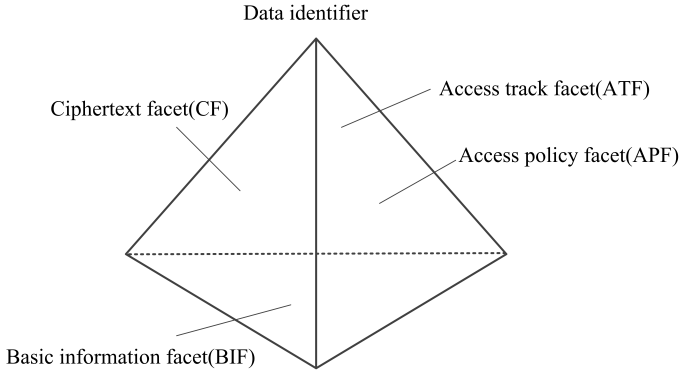
5.1 Tetrahedron model for protected data

In the self-contained data access control mechanism, data has much more information during its life time, such as security and tracks of data access history. Also, different types of information belonging to the data requires different corresponding operations. In order to describe the encrypted data in cloud computing completely and determine the required operations based on these encrypted data, we proposes a data model which is called Tetrahedron Model for Protected Data (TMPD). The model is composed of a vertex, four facets, as shown in Figure 1. The vertex represents the identifier for the protected data; the bottom facet represents the basic information of data; the three side facets represent the cipher data, the access tracks information and the access policy.

Firstly, we give the definition of the data structure of the protected data. Then, we give the operations of the data.

Definition 3. The structure of the tetrahedron model for protected data.

Figure 1 The structure of tetrahedral model for protected data.



The tetrahedral model is composed of a vertex and four facets, and is described by a 5-tuple:

$$CipherDataTetrahedron = \{V, CF, APF, ATF, BIF\} \tag{12}$$

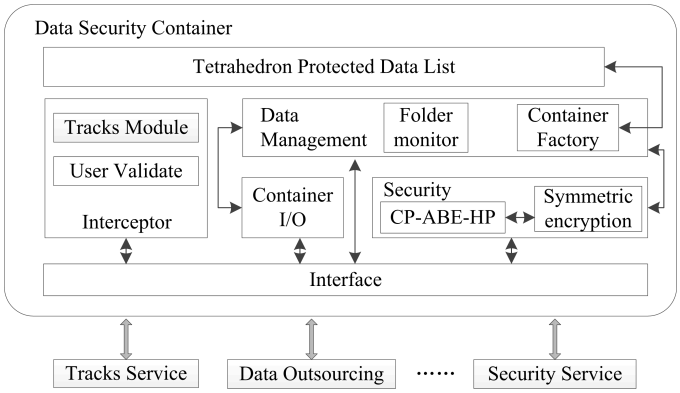
Where:

- V is the vertex of the tetrahedron and uniquely identifies a tetrahedron.
- CF (ciphertext facet) is the facet that describes the ciphertext of data and the symmetric encryption key.
- ATF (access tracks facet) is the facet that describes the tracks of the data access history: the read/write tracks and obtaining tracks of the ciphertext.
- APF (access policy facet) is the facet that describes the access policy which is encrypted.
- BIF (basic information facet) is the facet that describes the basic information of data, such as owner, size, create time and last modified time.

For each facet, except for the basic get/set operations, the TMPD also defines the following data operations.

- For the vertex of the TMPD, query operations by the unique identifier of the data are provided.
- For ciphertext facet, the symmetrical and asymmetrical encrypt/decrypt operations are provided.
- For access policy facet, only symmetrical encrypt/decrypt operation, which is needed when updating the read/write tracks of the data, are provided.
- For access tracks facet, two kinds of recording operations: security recording and general recording are provided. The security recording operation records the tracks with encrypt protection. The general recording operation records the obtaining tracks.
- For basic information facet, the read/write/update operations for every attributes of the basic information are provided.

Figure 2 The architecture of data security container.



5.2 Data security container architecture

The self-contained data access control mechanism uses hybrid encryption method to achieve security control. It also ensures the efficiency of encryption and decryption of large data. That is to say, we encrypt the data by symmetric encryption scheme, and the symmetric key is one-time key and generated randomly. Then we use CP-ABE-HP scheme to encrypt the symmetric key to achieve security and access control.

As the tetrahedral model described in section 5.1, there are series of data operations for the tetrahedral data. It is necessary to encapsulate the data operations with specific interface for users. Inspired by the essence of the web container and EJB container, where the container provides a collection of essential functions for managing and manipulating the components, we propose the Data Security Container (DSC) architecture to meet our requirements, as shown in Figure 2.

The components of DSC architecture mainly includes:

- Tetrahedron protected data list is a collection structure which is mainly used to store tetrahedron protected data. Meanwhile, the structure can be mapped to a folder in the user's file system.
- Data management module is used for managing the local and shared tetrahedron protected data. The folder monitor is responsible for monitoring the file operation events, such as create event, modify event and delete event, in the specific folder of user file system. And the container factory is primarily used to operate the tetrahedron protected data list by calling security modules.
- Security module is used to provide the security service. Symmetric encryption is responsible for encrypting data by one-time key which is generated randomly. The CP-ABE-HP module is mainly used to encrypt the asymmetric encryption key, so that it can meet the requirements of data access control.
- Interceptor includes a tracks interceptor which is used for recording the read/write tracks, and a user validation interceptor used for checking the legitimacy of user identity.
- Container I/O module is used for providing data outsourcing service.

With the mainly components, the DSC can provide the encryption/decryption service, access control service, tracks service, data outsource service and so on.

5.3 Easy Share system

Based on the CP-ABE-HP scheme and the self-contained data protection mechanism, we have implemented a tool named Easy Share(ES). The architecture of ES is shown in Figure 3. The system mainly includes five parties:

- Trust center including PKG and AA, which is responsible for initialization and key generation of the CP-ABE-HP scheme and IBE scheme(protecting the data transfer).
- Cloud server provider(data center), which is used for private data storage or data sharing.
- Data security container, which is introduced in section 5.2.
- User client, which is responsible for interaction with the user.
- Control center, which is used for administrator of the system to manage the trust center and data center.

ES system needs to be initialized by distributing the public parameters to users. If a user wants to share his data with a group of specific users or store his data in the cloud storage environment, he just needs to encrypt the data with the access policy by the DSC, and then shares the data to the cloud. When the recipient gets the encrypted data from the cloud, he should firstly acquire the private key from the security server by the DSC. The security server contains two parts, the Attribute Authority(AA) and the PKG. AA firstly authenticates the recipient's attributes, and then asks PKG to generate the private key containing these authenticated attributes for the recipient, and lastly AA sends the private key to the recipient. If the recipient's attributes in his private key satisfy the access policy which is corresponding to the ciphertext, he can decrypt the data successfully. Otherwise, he can't decrypt it.

An implementation result is shown in Figure 4, the Figure 5(a) is the UI of the client and the Figure 5(b) is the UI of the system control center.

With the ES tool, we can store or share our data easily and safely in the environment of the cloud computing. When the user first uses the tool, he should set the default monitor folder in user file system, default access policy which is used for data encryption and other default setting. Then the tool could be running at background silently. If the user wants to share a file, he only needs to put the file into the monitor folder, and the tool will encrypt the file with default access policy and share the protected data automatically. When the recipient wants to use the shared protected data, he only needs to select the data in the cloud and download it by the tool. If the recipient's attributes satisfy the access policy, the tool will decrypt the protected data to the default folder. Meanwhile, the tool can record or view the read/write tracks of the data automatically. By this way, the tool can reduce the user's burden on data security control without complex operations. Also these encryption, decryption and tracks recordation processing are executed automatically at the background; hence, these data security functions are transparent to the users. The control center is designed using the Browser/Server architecture which is convenient for the administrator, and its responsibility is to manage the PKG server, AA server and data server.

In summary, the Easy Share system is an efficient tool for data sharing, which greatly reduces the operation burden of the end users.

Figure 3 The framework of Easy Share system.

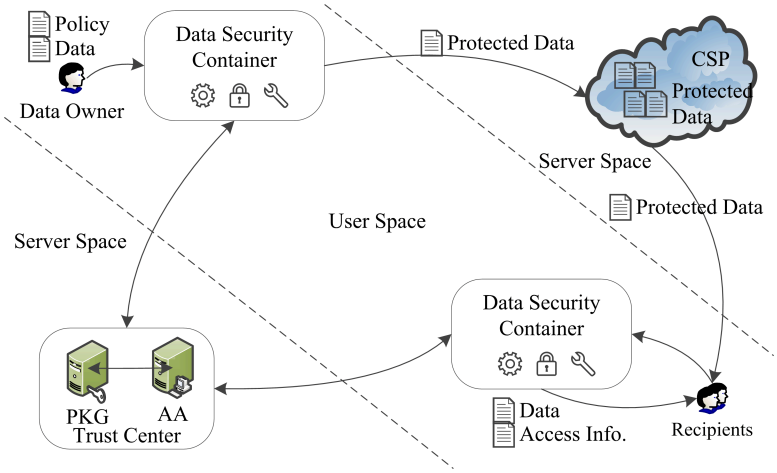
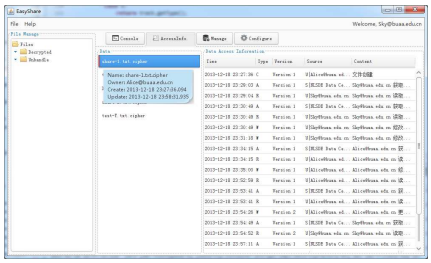
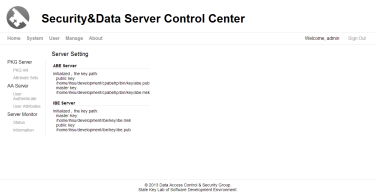


Figure 4 The implementation of the Easy Share system.



(a) The user client.



(b) System control center.

6 Conclusion

In CP-ABE schemes, policy hidden is of great significance in certain applications for protecting the privacy information of data provider and receiver. By introducing random element of subgroups into the policy key components, and with the property of subgroup element's orthogonal in composite order bilinear groups, the paper proposed the CP-ABE-HP scheme, which effectively realizes policy hidden in encryption. Meanwhile the tree-based access structure of CP-ABE-HP ensures that users can define their policies flexibly. Our scheme has very fewer extra costs of encryption and decryption compared with the CP-ABE schemes with tree-based access structure, and it has the Chosen-plaintext Attack security under the standard model. Also, inspired by the existing concept of container, the paper proposes the tetrahedron model for protected data and data security container to achieve the self-contained data protection mechanism efficiently. We implemented the mechanism and build a tool named Easy Share. The running results of Easy Share show that the mechanism simplifies the operations of users, and ultimately provides users with friendly, efficient, and secure data access experiences. For future work, we will try to apply CP-ABE-HP to some specific cloud storage environments.

Acknowledgement

References

Balu, A. and Kuppusamy, K. (2010a). Ciphertext policy attribute based encryption with anonymous access policy. *arXiv preprint arXiv:1011.0527*.

Balu, A. and Kuppusamy, K. (2010b). Privacy preserving ciphertext policy attribute based encryption. In *Recent Trends in Network Security and Applications*, pages 402–409. Springer.

Beimel, A. (1996). *Secure schemes for secret sharing and key distribution*. PhD thesis, PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel.

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 321–334. IEEE.

Boneh, D., Goh, E.-J., and Nissim, K. (2005). Evaluating 2-dnf formulas on ciphertexts. In *Theory of cryptography*, pages 325–341. Springer.

Cheung, L. and Newport, C. (2007). Provably secure ciphertext policy abe. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 456–465. ACM.

Doshi, N. and Jinwala, D. (2012). Hidden access structure ciphertext policy attribute based encryption with constant length ciphertext. In *Advanced Computing, Networking and Security*, pages 515–523. Springer.

Frikken, K., Atallah, M., and Li, J. (2006). Attribute-based access control with hidden policies and hidden credentials. *Computers, IEEE Transactions on*, 55(10):1259–1270.

Goyal, V., Jain, A., Pandey, O., and Sahai, A. (2008). Bounded ciphertext policy attribute based encryption. In *Automata, Languages and Programming*, pages 579–591. Springer.

Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. ACM.

Ibraimi, L., Tang, Q., Hartel, P., and Jonker, W. (2009). Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In *Information Security Practice and Experience*, pages 1–12. Springer.

Kapadia, A., Tsang, P. P., and Smith, S. W. (2007). Attribute-based publishing with hidden credentials and hidden policies. In *NDSS*. Citeseer.

Lai, J., Deng, R. H., and Li, Y. (2011). Fully secure cipertext-policy hiding cp-abe. In *Information Security Practice and Experience*, pages 24–39. Springer.

Lai, J., Deng, R. H., and Li, Y. (2012). Expressive cp-abe with partially hidden access structures. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pages 18–19. ACM.

Li, J., Ren, K., Zhu, B., and Wan, Z. (2009). Privacy-aware attribute-based encryption with user accountability. In *Information Security*, pages 347–362. Springer.

Li, X., Gu, D., Ren, Y., Ding, N., and Yuan, K. (2012). Efficient ciphertext-policy attribute based encryption with hidden policy. In *Internet and Distributed Computing Systems*, pages 146–159. Springer.

Nishide, T., Yoneyama, K., and Ohta, K. (2008). Attribute-based encryption with partially hidden encryptor-specified access structures. In *Applied cryptography and network security*, pages 111–129. Springer.

Sahai, A. and Waters, B. (2005). Fuzzy identity-based encryption. In *Advances in Cryptology–EUROCRYPT 2005*, pages 457–473. Springer.

Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11):612–613.

Waters, B. (2005). Efficient identity-based encryption without random oracles. In *Advances in Cryptology–EUROCRYPT 2005*, pages 114–127. Springer.

Waters, B. (2011). Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography–PKC 2011*, pages 53–70. Springer.

Yu, S. (2010). *Data sharing on untrusted storage with attribute-based encryption*. PhD thesis, WORCESTER POLYTECHNIC INSTITUTE.

Yu, S., Ren, K., and Lou, W. (2008). Attribute-based content distribution with hidden policy. In *Secure Network Protocols, 2008. NPSec 2008. 4th Workshop on*, pages 39–44. IEEE.

Yu, S., Ren, K., and Lou, W. (2010). Attribute-based on-demand multicast group setup with membership anonymity. *Computer Networks*, 54(3):377–386.