An Access Control Scheme Based on Access Tree Structure Pruning for Cloud Computing

Ze Wang, Minghua Gao, Lu Chen, and Shimin Sun
(Corresponding author: Ze Wang)

School of Computer Science and Technology, Tianjin Polytechnic University
399 Binshui W Rd, Xiqing, Tianjin, China
(Email: wangze@tjpu.edu.cn)

(Received July 8, 2019; Revised and Accepted Dec. 6, 2019; First Online Apr. 19, 2020)

Abstract

In the era of rapid development of information, people are spreading and sharing information all the time. These resources bring us a lot of privacy challenges while bringing us convenience. Therefore, We propose an attribute based encryption access control scheme based on access tree structure pruning (ATSP-ABE). The scheme mainly prune the branch of the ID attribute of the right subtree user managed by the Data Owner (DO) and design the permission access attribute as the leaf node to replace the branch. For the left subtree of the access tree managed by the Attribute Authorization Center (AAC) the decision tree is generated by the data of the user feature attribute and the subtree with the best pruning performance is selected as the pruning result. Finally, pruning the reduced feature attributes in the decision tree in the access left subtree. The experimental results show that the ATSP-ABE scheme can improve the computational efficiency of attribute-based access control encryption, decryption and user attribute revocation in cloud computing. More than that makes the access tree structure more concise and strengthen the DO control attribute ability. Reducing Calculation overhead in the process of encryption and decryption of DO and AAC.

Keywords: Access Control; Attribute Based Encryption; Cloud Computing; Decision Tree

1 Introduction

In this new digital era filled with vast amounts of data or information, everyone enjoys the convenience of instant information sharing and dissemination. More and more user data is uploaded to third-party cloud servers for storage, management or exchange [8, 10]. In order to reduce the efficiency of user data transmission, an access control scheme is introduced in the face of privacy protection and access threats. Access control scheme [6, 11, 19] be used to protect personal data on public platforms from unauthorized access or disclosure. In addition, data encryption algorithms [27] are often used to prevent platform operators and other attackers who are curious during data communication from snooping. Therefore, an efficient access control scheme must be designed according to the requirements of the user to make the stored data value of the shared user available to the authorized user. Once the user's data is outsourced to the cloud, the user will loses Control ability of their data. Since access control schemes are becoming more and more important. In order to solve the data protection problem in cloud storage, attribute-based encryption(ABE) [13] is considered one of the most promising technologies, which is from identity-based passwords. Learning from the development of [23], the ABE scheme performs fine-grained access control on the data stored in the cloud server by setting attributes. The ABE scheme mainly consists of the following two types, such as Ciphertext-Policy Attribute-Based Encryption(CP-ABE) [7, 31, 32] and Key-Policy Attribute-Based Encryption(KP-ABE) [9, 18, 21]. Considering the frequent update of ciphertext and a large number of users of the mobile Internet, CP-ABE is more suitable for fine-grained data access control in cloud storage scenarios.

The access structure that CP-ABE can adopt is a linear access structure or a tree access structure. Waters *et al.* [26] implements a CP-ABE access control scheme based on linear access structure to support attribute revocation. Linear access structures are better able to solve completely independent properties, but less efficient for incompletely independent properties that appear in real-world situations. Xiong and Simoes and Touatil *et al.* [24, 25, 29] use the tree access structure to implement the CP-ABE access control scheme to support attribute revocation, which solves the problem of the user's incomplete independence attribute in the actual situation. Huang *et al.* [22] proposes a multi-authority revocable attribute-based encryption (MA-ABE) scheme, which the classifification manages user attributes, to relieve the management burden of single organization effectively. In addition, the tree access policy and the secret

sharing scheme are adopted to implement fine-grained access control of shared information and support system attribute revocation. In the attribute-based encryption-based access control scheme in the cloud computing environment [1, 15, 28], leaf nodes in the access tree represent user attributes, and non-leaf nodes represent access policies.

The AAC determines whether the user satisfies the access tree structure through user attributes, thereby being able to manage and control the users who want to access the data resources. However, all user attribute data is managed and controlled by a third party, so DO loses the authority to manage its shared data. Yang *et al.*proposed a scheme [16] to divide the user attribute into two parts, which are managed and controlled by AAC and DO respectively, DO also has the right to manage its attributes. However, as the number of users and attributes increases, the workload of AAC and DO increases, resulting in reduced efficiency. Therefore, based on the literature [16], this paper optimizes the structure of the access subtree managed by AAC and DO respectively, and improve the efficiency of user attribute management and control. By simplifying the access tree, the computational overhead of AAC and DO is reduced. Through the experimental results, it can be verified that the pruning of the left subtree is constructed, the decision tree [12, 20, 30] is constructed to process the user data, and the subtree with the maximum pruning performance can be significantly improved. Then, the ATSP-ABE scheme proposed in this paper can also implement the function of user attribute revocation [2, 3, 17]. Compared with the two ABE schemes using linear access structure [26] and tree access structure [16], the proposed scheme greatly improves the performance of private key generation, password text size, encryption and decryption, and user attribute revocation in cloud computing.

2 Our Contribution

1) An access control scheme based on access tree structure pruning in cloud computing environment is proposed. It performs different pruning on the access right subtree and access left subtree which is managed by DO and AAC respectively. Accessing the right subtree to prun the branch where the user ID attribute node is located, and design the permission access attribute to replace the branch with the leaf node. This scheme allows DO to retain the key attributes of its shared data, fully control its shared data and reduce the computational overhead of DO in the access policy.

2) Accessing the left subtree in the cloud computing environment first generates a decision tree based on the data of the user feature attribute, which selects the subtree with the best pruning performance as the pruning result. Finally, accessing the left subtree will reduce the feature attributes in the decision tree. After pruning, the access tree is simplified. The solution reduces the computational overhead of the AAC in the access policy and improves the management and control efficiency of the AAC.

3) An efficient cloud computing access control scheme based on CP-ABE structure is proposed. Users can decrypt ciphertext and attributes with a small amount of calculation. In our scheme, it can be achieved by pruning, which only a small amount of computational overhead is required. The effectiveness of the scheme is demonstrated by comparison with other schemes in terms of computational complexity and communication overhead.

2.1 System Model

In the architecture of ATSP-ABE scheme, there are four entities (see Figure 1): Data Owner (DO), Data User (DU), Platform Server (PS), and Attribute Authentication Center (AAC). The DO uploads the algorithm encrypted file to the PS, and the security of the file is guaranteed by the access control and decryption process. PS and AAC always stay online, assuming it has infinite storage and computing power to ensure that DU download and decrypt these files from PS. AAC is responsible for the distribution and revocation of attributes, then the attributes of the user are jointly controlled by AAC and DO. AAC is responsible for managing the user's feature attributes, which is a set of attributes describe the DU. We assume that DO not only stores data files, but also creates a set of attribute-defined access policies for its data files.

The complexity of the ATSP-ABE algorithm in the cloud environment is proportional to the complexity of the structure of its corresponding access tree. Therefore, the algorithm delivers a lot of mixed content instead of the core algorithm to the PS in the implementation process. Most of the attributes of the DU are managed and controlled by AAC. The DO still retains its key attributes and shared data calculations, maintains its original security, the security level and the locally calculated security level in the original CP-ABE scheme remain unchanged. During the encryption and decryption operations, the PS has access to most of the keys in the tree structure but not all keys. In the process of decrypting the calculation, the bilinear pair in the ciphertext and key generation spend a lot of calculations in the system, so we safely pass this part of the operation to the PS. The last step of the decryption operation is performed by the data user DU itself, and the data sharing resources will not leak to the PS.

3 Detailed Description

The traditional CP-ABE access control scheme mainly uses the access tree as the access policy. The complexity of the access tree determines the efficiency of DO and

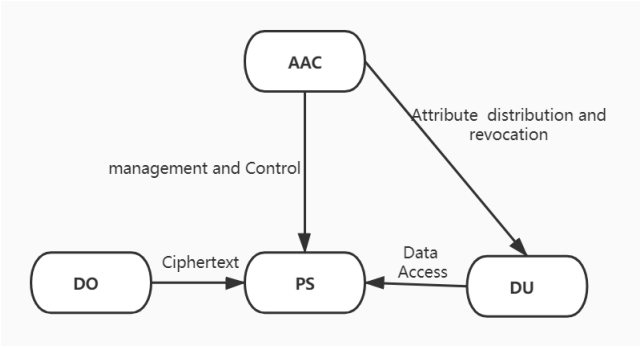

Figure 1: The proposed system model

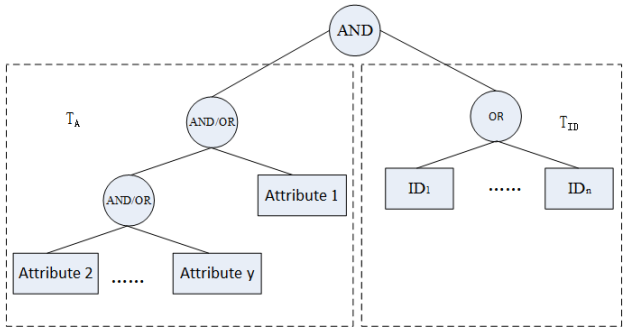

Figure 2: The access tree structure of the CP-ABE

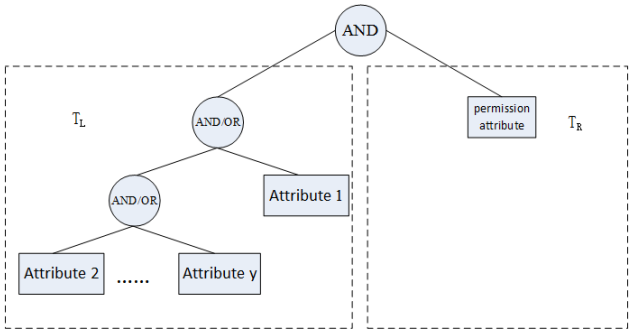

Figure 3: The access tree structure of the ASTP-ABE scheme

AAC access control for user attributes. Therefore, the access control scheme is designed for pruning the access tree structure in this paper, which simplifies the access tree and reduces the complexity. The access tree structure is divided into accessing the left subtree T_A and accessing the right subtree T_{ID}, which contain different attributes. The former contains the feature attributes of the DU, and the latter contains the ID attributes of the DU. Therefore, our proposed ATSP-ABE scheme prunes the two subtrees of these schemes separately.

3.1 The Pruning of the Access Right Subtree

The Figure 2 depicts the access tree structure of the CP-ABE scheme in literature [16], and $\{A_1, A_2, ..., A_y\}$ representing a user's feature attribute set. $\{ID_1, ID_2, ..., ID_n\}$ representing the user's ID attribute collection. The access tree T is a binary tree with access to the left subtree T_A and access to the right subtree T_{ID}. The feature attribute in the access left subtree is managed by AAC, and the user ID attribute in the access right subtree is controlled by DO. The root node of the access tree is an "AND" node, which indicates that the user attribute must satisfy both the access left subtree and the right subtreeto satisfy the access policy. Accessing the right subtree contains an "OR" node whose leaf nodes are related to the user ID attribute. In this access tree structure, although most user attributes are managed and controlled by AAC.

The ID attribute of many users managed by DO still affects the efficiency of user attributes and key control. Therefore, in order to make the data access more convenient and faster, this paper designs the structure of accessing the right subtree to reduce the computational cost of access control during encryption and decryption. Experiments show that the pruning result of accessing the right subtree is shown in Figure 3. The structure is simplified and the computational overhead is reduced, and attribute revocation can be implemented more efficiently. The branch accessing the right subtree in the CP-ABE scheme is an "OR" node and n ID attribute nodes. Each attribute node will be assigned a separate user, which will cause a burden on the key calculation. When the algorithm needs to determine the access tree structure, the "OR" node is used as the root node in the right subtree to prun the branch, including its n ID attribute nodes and is designed to access the license access attribute of the right subtree to replace the branch. The changed access to the right subtree allows the security of encryption and decryption to be guaranteed, since the last step of the decryption operation is performed by the data user DU itmself. Permission access properties are constantly updated to ensure the security of shared data. At the same time, the pruning of the access tree also reduces the computational cost of encryption and decryption. It enables users to access the data they need more effectively and to share data easily.

3.2 The Pruning of the Access Left Subtree

The AAC manages and controls many feature attributes of DU. Because of the overlapping feature attributes among DUs, AAC performs many tasks to repeatedly determine feature attributes which increases the workload of AAC. Therefore, we use the decision tree [14] to classify user data, we can divide user data into different categories. Depending on the category of the DU, users can be granted different permissions. In this way, AAC no longer needs to perform access control defined by one DU after another. Instead, it firstly determines the classification of DU data and then performs access control defined by the attributes of the data classification which effectively reduces the efficiency and computational overhead of AAC management and controls the attributes.

We have adopted the idea of reducing the error pruning method (REP) [4,20]. It uses a separate data set to make up for the ERP of the pruning process, which not only considers the accuracy of the classification but also considers the performance of both the classification balance and the complexity. In the case of ensuring the accuracy of the algorithm, the decision tree is simplified and the computational complexity of the access tree structure is reduced. In Table 1, we list the symbols used in ATSP-ABE:

1) Accuracy of classification. It mainly reflects the classification accuracy of decision trees, defined as:

$$m(T') = \frac{1}{Q'} \sum_{i \in Y} b(i)'.$$

We use it to calculate the classification accuracy of the decision tree, which is given by the ratio of the sum of the correct number of users per node to the sum of the number of users in the pruning set. If the classification accuracy of the decision tree is greater, the accuracy of the decision tree will higher.

2) The balance of classification. It mainly reflects the classification balance of the decision tree, defined as:

$$r(T') = \frac{1}{Q'} \sum_{i \in \omega} q(i)' r(i).$$

Where r(i) is the classification accuracy of the node and can be calculated as:

$$r(i) = \begin{cases} \frac{m(i)'}{m(i)} & m(i)' < m(i) \\ \frac{m(i)}{m(i)'} & m(i)' > m(i) \end{cases}$$

$m(i)$ and $m(i)'$ in the above formula are mainly calculated by the pruning set and the training set, and can be defined as:

$$m(i)' = \frac{b(i)'}{q(i)'}$$

$$m(i) = \frac{b(i)}{q(i)}$$

Therefore, the larger the classification balance value of the decision tree, the higher the classification stability of the entire decision tree.

3) Complexity. If the decision tree is too complex, the number of users arriving at certain nodes will be reduced, making the decision tree unable to process its user set. In order to ensure the accuracy of classification and the performance of classification, the complexity of the decision tree should be reduced as much as possible. The combination of leaf nodes and depths of the decision tree can be represented by t. Among $t = \omega + \upsilon$ its complexity can be defined as follows:

$$f(t) = \begin{cases} 0 & t < 4 \ or \ t > 35 \\ \frac{t+10}{20} & 4 \le t \le 10 \\ \frac{44-t}{40} & 20 \ge t > 10 \\ \frac{50-t}{30} & 35 \ge t > 20 \end{cases}$$

In summary, the classification between the number of leaf nodes and the depth of the tree in the decision tree is best. When the range of t is $t < 4 \ or \ t > 35$, it is a very unfavorable situation. This paper proposes to use the pruning performance to evaluate the performance of the decision tree, as can be defined as follows:

$$P(T') = x_1 m(T') + x_2 r(T') + x_3 f(T').$$

In the above formula x_1, x_2, x_3 represents the classification accuracy, classification balance, and complexity ratio of the decision tree, at the same time, satisfies $x_1 + x_2 + x_3 = 1$. In short, we allocate the ratio of the three performances evenly, and can also be based on different actualities. In the case of the proportion of each performance is assigned. During the pruning process, the pruning performance of each candidate subtree is compared, the pruning tree with the highest pruning performance is selected to ensure the optimal pruning performance of the decision tree.

3.3 The Detailed Trimming Process

1) From the bottom up, each subtree in the decision tree is a candidate subtree of the pruning, the subtree is replaced by the leaf node, and the node is identified by the category represented at most instances, reaching the leaf node in the training set. It generates a set of pruned subtrees $\{T'_0, T'_1, ..., T'_i\}$ representing the decision trees that T'_0 have not been pruned.

2) Based on the categorical data of the trained set and the data of the pruned set of the original decision tree, we calculated the pruning performance $P(T'_i)$ of each pruned subtree T'_i.

3) In the candidate subtree set $\{T'_0, T'_1, ..., T'_i\}$, the algorithm compares the pruning performance $P(T'_i)$ of each candidate subtree and selects the tree with the highest pruning performance as the final decision tree.

4) The pruning of the decision tree subtree corresponds to the pruning of the feature attributes, and the reduction of each subtree corresponds to the reduction of the determined feature attributes. Finally, the algorithm prunes the reduced characteristic attributes of the decision tree on the access tree structure.

5) By separately pruning the left and right subtrees of the access tree, the access tree structure is simplified, the computational overhead of DO and AAC

Table 1: Notations for ATSP-ABE

$m(T')$	the taxonomy veracity of the decision tree T'.
$r(T')$	the taxonomy balance of the decision tree T'.
$f(T')$	the complexity of the decision tree T'.
$r(i)$	the taxonomy balance of the node i.
$m(i)'$	the taxonomy veracity of the node i in the pruned set.
$m(i)$	the taxonomy veracity of the node i in the trained set.
Q'	the users number in the pruned set.
Q	the users number in the trained set.
$q(i)'$	the users number of the node i in the pruned set.
$q(i)$	the users number of the node i in the trained set.
$b(i)'$	the users number belongs to the node i in the pruned set.
$b(i)$	the users number belongs to the node i in the trained set.
ω	the number of the leaf node in the decision tree.
υ	the depth of the decision tree T'.
i	the i-th node of the decision tree.

management and control attributes is realized. Improve the efficiency of its property management and control.

4 The ATSP-ABE Algorithm

The mathematical basis of the ATSP-ABE algorithm is bilinear mapping. Let q be a large prime number, G_1 and G_2 be two multiplicative cyclic groups of order q, p is the generator of G_1, and $e : G_1 * G_1 \to G_2$ is a bilinear map. It has the following properties:

- The Bilinear. For any $P, Q, R \in G_1$ and $a, b \in Z_q$, there are:

$$\begin{aligned} e(P \cdot Q, R) &= e(P, R)e(Q, R) \\ e(P^a, Q^b) &= e(abP, Q) = e(P, abQ) = e(P, Q)^{ab} \end{aligned}$$

- The Non Degeneracy. There is $P, Q \in G_1$ that makes $e(P, Q) \neq 1$, among them, 1 is the generator of the multiplicative cycle group G_2.

- The Computability. For all $P, Q \in G_1$, there is a valid algorithm for the calculation of $e(P, Q)$.

The ATSP-ABE algorithm consists of four parts. They are system settings, encryption algorithms, user private key generation and decryption algorithms, described as follows:

System setting. The algorithm selects a bilinear multiplicative cyclic group G_1 with a prime order q and a generator p. Let $e : G_1 * G_1 \to G_2$ represents a bilinear map. The AAC selects two random parameters $a_1, b_1 \in Z_q$, and generates the first master key as:

$$MK_1 = \{b_1, p^{a_1}\}.$$

The first public key is as:

$$PK_1 = \{G_1, p, \eta_1 = p^{b_1}, e(p, p)^{a_1}\}.$$

The DO selects two random parameters $a_2, b_2 \in Z_q$, and generates the second master key as:

$$MK_2 = \{b_2, p^{a_2}\}.$$

The second public key is as:

$$PK_2 = \{G_1, p, \eta_2 = p^{b_2}, e(p, p)^{a_2}\}.$$

The system setting also selects a random parameter $\varepsilon_0 \in Z_q$ for later use.

Encryption algorithm. In the case of access tree structure, ATSP-ABE algorithm encrypts information M. Select two random parameters $\mu_1, \mu_2 \in Z_q$ and select two polynomials $g_L(x), g_R(x)$ to represent access the left subtree and the right subtree respectively. Suppose that the leaf node set of accessing the left subtree is W, which A_λ is the permission access attribute node for accessing the right subtree. The $att(w)$ is a function of the attribute that w is a leaf node and is associated with a leaf node X in the access tree. The algorithm uses a hash function $H : \{0, 1\}^* \to G_1$, and describes any attribute on a bilinear map as a binary string of random elements. The access tree generates the ciphertext as:

$$\begin{aligned} CT = \{&T_L, \tilde{C} = Me(p, p)^{a_1,\mu_1} e(p, p)^{a_2,\mu_2}, \\ &C_1 = \eta_1^{\mu_1}, C_2 = \eta_2^{\mu_2}, \\ &\forall w \in W : C_w = p^{g_{w(0)}}, C'_w = H(att(w))^{g_{w(0)}}, \\ &\forall \lambda \in A_\lambda : C_\lambda = p^{\mu_2}, C'_\lambda = H(att(w))^{\mu_2}\}. \end{aligned}$$

User private key generation. Including the private key of the feature attribute and the private key of the license attribute, respectively calculated as follows: The AAC algorithm inputs the attribute set A_u of the user u, and generate a key for the attribute set. The algorithm selects a random number $\varepsilon \in Z_q$ and $\varepsilon_j \in Z_q$ and selects the random number $j \in A_u$ for

the feature attribute. The feature attribute private key is as follows:

$$SK_1 = \{D_1 = P^{(a_1+\varepsilon)/b_1}, \\ \forall j \in A_u : D_j = p^{\varepsilon} \times H(j)^{\varepsilon_j}, D'_j = p^{\varepsilon_j}\}.$$

The data owner DO's algorithm enters the user's permission access attribute and outputs the private key of the license access attribute. The algorithm selects a random number $\varepsilon_0 \in Z_q$ and $\varepsilon_\lambda \in Z_q$ for the user. The private key of the license access attribute is as follows:

$$SK_2 = \{D_2 = P^{(a_2+\varepsilon_0)/b_2}, \\ D_\lambda = p_0^{\varepsilon} \times H(att(\lambda))^{\varepsilon_\lambda}, D'_\lambda = p^{\varepsilon_\lambda}\}.$$

It then executes the above two-part algorithm, and generates the user private key as:

$$SK = (SK_1, SK_2).$$

Decryption algorithm. Including decryption access to the left subtree and the right subtree. The first part is the decryption process of accessing the left subtree, which is the same as the CP-ABE algorithm and represtents by A_1.

$$A_1 = DecryptNode(CT, SK_1, w) \\ = e(p,p)^{\varepsilon\mu_1}.$$

$$M_1 = Decrypt(CT, SK_1) \\ = \tilde{C}/(e(C_1, D_1)/A_1) \\ = \tilde{C}/e(p,p)^{a_1,\mu_1} \\ = Me(p,p)^{a_2,\mu_2}.$$

The second part is the decryption process of accessing the right subtree and represtents by A_2. Finally we can get the ciphertext shown as below:

$$A_2 = DecryptNode(CT, SK_2, \lambda) \\ = \frac{e(D_\lambda, C_\lambda)}{e(D'_\lambda, C'_\lambda)} \\ = \frac{e(p^{\varepsilon_0} \times H(att(\lambda))^{\varepsilon_\lambda}, p^{\mu_2})}{e(p^{\varepsilon_\lambda}, H(att(\lambda))^{\mu_2})} \\ = e(p,p)^{\varepsilon_0,\mu_2}.$$

$$M_2 = M_1/(e(C_1, D_2)/A_2) \\ = M_1/e(p,p)^{a_1,\mu_2} \\ = M.$$

5 Security Proof

The security analysis is mainly composed of the following four aspects: full control of shared data, prevention of user key leakage, untrusted third party organizations, and data confidentiality analysis.

Full control of shared data: With the help of the license access properties, the data owner DO can fully manage their data resources. The branch in the right subtree in the CP-ABE scheme is the OR node and its n ID attribute nodes, which are assigned to each user and easily burden the key calculation. When the algorithm needs to determine the access tree structure, branches with the "OR" node as the root node in the right subtree, including its n ID attribute nodes, are pruned, and the branch is designed for the right subtree design permission access attribute. This can not only meet the data owner DO control shared data requirements, but also reduce the computational overhead and improve the efficiency of the ATSP-ABE access control scheme.

Preventing user key abuse: This problem is solved by means of user private key separation, and the data owner DO can immediately suspend any user sharing data. The data owner DO still controls the final step of decrypting the ciphertext. The license access attribute is continuously updated to ensure the security of shared data. At the same time, access tree pruning also reduces the computational overhead of encryption and decryption, enabling users to access required data more efficiently and quickly, and easy to implement data sharing.

untrusted third party organizations: The data owner DO can entrust a semi-trusted organization to complete the revocation task. The data owner DO personal information is based on the proxy re-encryption method and is transparent to the organization's transmission. The user attribute undo operation may revoke one or more attributes owned by the user without affecting the current attributes of other users. The revocation will cause a large number of key update operations, and the user and ciphertext encrypted with the expired public key need to be updated. This paper uses the platform server PS or other service system to solve the problem. When the user attribute revocation occurs, the attribute authorization center AAC passes the information to the platform server PS to directly revoke the user attribute. When the user wants to access the encrypted data, the platform server PS firstly checks the attributes of the user. If the platform server PS determines that the user's attribute does not satisfy the access policy, the user will not be assigned an encrypted data key.

Data confidentiality: The data confidentiality is analyzed by the following scheme, which proves that the scheme satisfies the indiscernibility of the message under the assumption of DBDH. The mathematical basis of the security analysis is the Decision Bilinear key exchange algorithm DBDH [5](Decisional Bilinear Diffie-Hellman) hypothesis. It is assumed that $\alpha, \beta, \gamma, z \in Z_q$ is uniformly selected and G_1 is a group

whose prime order q and the generator element is p. The DBDH assumption means that an attacker cannot distinguish tuples $(p^\alpha, p^\beta, p^\gamma, e(p,p)^{\alpha\beta\gamma})$ and $(p^\alpha, p^\beta, p^\gamma, e(p,p)^z)$ in a polynomial time with a non-negligible advantage.

Theorem 1. *Under the security model of DBDH hypothesis, if an attacker can destroy the model of the algorithm with a non-negligible advantage, and then we can construct a simulator to solve the DBDH problem.*

Proof. Suppose that attacker I can attack the algorithm's model in a polynomial time with a non-negligible advantage ε. We set up a simulator E, which can perform the DBDH match with the advantage $\varepsilon/2$. The simulator E is set to:G_1, G_2 is a valid bilinear map e with generator p and DBDH instance $(p^\alpha, p^\beta, p^\gamma, e(p,p)^z)$, where $Z = \alpha\beta\gamma$ or random. The simulator operates as follows: □

1) Initialization: Attacker I selects a challenged access structure T^* and sends it to Simulator E.

2) Setting: Simulator E sets the parameter $Y = e(A,B) = e(p,p)^{\alpha\beta}$ to select the random parameter $\delta \in Z_q$ and sets the parameter $\eta = p^\delta, \varphi = p^{1/\delta}$. Send public parameters to attacker I.

3) Search 1: The private key that the attacker I requested from the simulator E to set the attribute.

$$w_i = \{\alpha_i | \alpha_i \in \Omega \cap \alpha_i \notin T^* \cup U_\lambda\}.$$

Simulator E selects a random function F_{s_k} for the attribute authorization center AAC, It sets the parameters $y_{k,u} = F_{s_k}(\lambda)$, Where λ is the user permission access attribute. Selecting parameter $r, s_k \in Z_q$, setting parameter $D = p^{(y_k + y_{k,u} + r)/\delta}$. If attribute $\alpha_i \in w_i$, it sets parameters $D_{\alpha_i} = p^r H(\alpha_i)^{r_{\alpha_i}}, D'_{\alpha_i} = p^{r_{\alpha_i}}$, and sends the private key to attacker I.

4) Challenge: Attacker I submits a challenge attribute and two challenge messages M_0, M_1 to Simulator E. We assume that attacker I never asks for the private key in the Search 1 setting. The simulator E randomly selects $\beta \in \{0,1\}$, and encrypts the information as:

$$M_\beta : CT = (T^*, \tilde{C} = M_\beta Z, C = \eta^\alpha, \forall i \in T^* : C_i = p^{\alpha_i}, C'_i = H(i)^{\alpha_i}).$$

According to the encryption algorithm in the ATSP scheme, we set the root node τ for the challenge access tree T^*. The simulator E sends ciphertext to the attacker I. If $Z = e(p,p)^{\alpha\beta\gamma}$, we implicitly set $\tau = c$, that is $Y^\tau = Z = e(p,p)^{\alpha\beta\gamma}$ and $C = \eta^c, C_i = p^{c_i}$. It shows that the ciphertext is a valid random encryption of the information M. Otherwise, if $Z = e(p,p)^z$ for a random z, $\tilde{C} = M_\beta e(p,p)^z$. From the perspective of the attacker I, CT is a random element of G_2, and the ciphertext does not contain information M_β.

5) Search 2. It performs the same operation as Search 1.

6) Conjecture. The attacker I submits β' of the guess β. If $\beta = \beta'$, the simulator E will output 0, indicating $Z = e(p,p)^{\alpha\beta\gamma}$, otherwise the simulator will output 1 to indicate that the attacker I has not obtained any information of the encrypted M_β.

- If $\beta' \neq \beta$, then there is

$$P_r[\beta' \neq \beta | Z = (p,p)^z] = 1/2.$$

- If $\beta' = \beta$, we define the advantage of the attacker I as ε, and there is

$$P_r[\beta' = \beta | Z = (p,p)^{\alpha\beta\gamma}] = 1/2.$$

- Therefore, the advantage of the simulator E in the DBDH match is

$$\begin{aligned} Adv &= P_r[\beta' = \beta] - 1/2 \\ &= 1/2 \cdot (P_r[\beta' = \beta | Z = (p,p)^{\alpha\beta\gamma}] \\ &+ P_r[\beta' = \beta | Z = (p,p)^z]) - 1/2 \\ &= \varepsilon/2. \end{aligned}$$

Only when the attribute settings satisfy the access policy, the user can decrypt and get the encrypted information of the scheme. The ATSP-ABE model is proved to be safe by DBDH hypothesis theory.

6 Results and Discussion

6.1 Analysis of the Pruning Results of the Access Tree

The ATSP-ABE scheme by using the pairing-based cryptography PBC (Pairing-Based Cryptography) library version 0.4.18. The experiment in this paper was carried out by using a PC with a dual-core 3.1GHz, Intel Core i7-6500U CPU, 16GB RAM, and 64-bit Win10 operating system. The experimental data set of this paper is the blood transfusion service center data set in the UCI machine learning database. The data set is moderately sized, and the access tree is not overly complex, facilitating pruning and is suitable for simple and intuitive descriptions. The blood service center has 748 user data, including 5 attributes. The category attribute is category, and the remaining attributes are represented by A_1, A_2, A_3, A_4 respectively. Their meanings are as follows:Category indicates whether blood was donated in March 2007. A_1 indicates the number of months since the last donation. A_2 indicates the total amount of donations. A_3 indicates the total number of months of blood donation. A_4 indicates the total amount of blood donation (c.c.).
In this paper, the independent pruning dataset is used to randomly extract the user data, and 60.70% of the

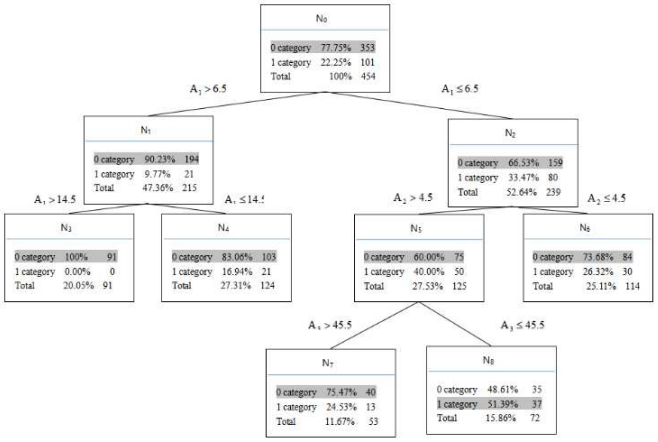

Figure 4: Training data set classification

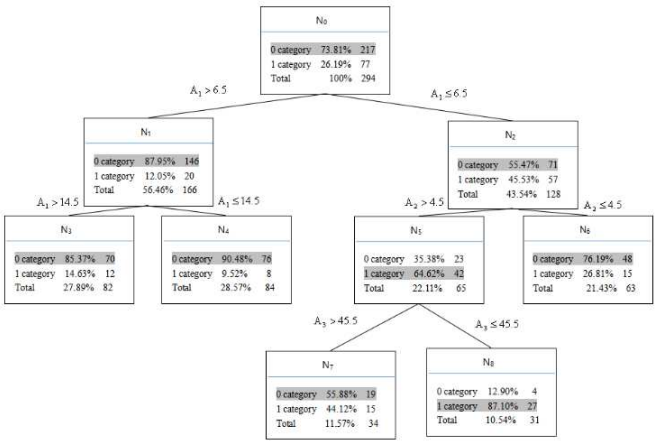

Figure 5: Pruning data set classification

users are selected as the training dataset, and the remaining 39.30% of the users are used as the pruning dataset. The decision tree generated by the user's feature attribute data is shown in Figure 4 and Figure 5. Figure 4 shows the classification of the training set, and Figure 5 shows the classification of the pruning set. Since the decision tree selected in this paper is not complicated, the sum of the leaf nodes and the depth of the tree is less than 10, which can reduce the proportion of the complexity of the decision tree when calculating the pruning performance. This paper adjusts the weight distribution to 45% 45% and 10%. The pruning performance is calculated as follows:

$$P(T') = 0.45m(T') + 0.45r(T') + 0.1f(T').$$

The nodes in the decision tree are represented as $N_t(t = 0, 1, 2, ..., n)$. The pruning performance of the subtree formed after removing the node N_i from the decision tree is expressed as $P(T'_{N_i})$. The classification accuracy of the nodes i on the pruning set and the training set can be obtained and as shown in Table I, and Table II shows the leaf nodes, classification accuracy $m(T')$, classification balance $r(T')$, tree complexity $f(T')$ and corresponding pruning performance $P(T')$ of candidate subtrees.

The algorithm uses these seven subtrees as candidate pruning subtrees and compares their pruning performance in bottom-up order. The candidate subtree has the best pruning performance, and the subtree is used as the final decision tree. The algorithm first prunes the branch from the original decision tree with node N5 and replaces it with a leaf node.The pruning branch N1 of the node is then replaced with a leaf node. The decision tree obtained using the REP pruning method is shown in Figure 6. The pruning performance method used in the paper obtains the decision tree after pruning, as shown in Figure 7. Compared to Figures 6 and 7, the decision tree generated by the pruning performance method used herein reduces the two leaf nodes compared to the REP pruning method. Reduce the size and complexity of the tree, making the structure of the access tree more concise and understandable.As shown in Table 4, the pruning performance method was 11.21% higher than the REP pruning method in the classification accuracy rate of category 1. While improving the classification accuracy, the goal of the access tree structure is more concise and the computational complexity is reduced.

The access left subtree T_A is composed of user feature attributes and is managed and controlled by the attribute authorization center AAC. The access right subtree T_ID consists of the attributes of the user data, and the last step of decryption is controlled by the data owner. In order to select the donors in March 2007 from the user data set and grant them special permissions, the access tree structure constructed using the CP-ABE algorithm is shown in Figure 8. The access tree structure expression is as follows: $("A_1 \leq 6.5" AND "A_2 > 4.5" AND "A_3 \leq 45.5" AND "A_4")AND("ID_1" OR "ID_2" OR ... "ID_{748}")$. Firstly, it is necessary to determine whether the feature attribute of the user satisfies the access to the left subtree. If the feature attribute of the user satisfies $("A_1 \leq 6.5" AND "A_2 > 4.5" AND "A_3 \leq 45.5" AND "A_4")$ the four attributes at the same time, the user feature attribute satisfies the left subtree. It is necessary to determine whether the user ID attribute is satisfied $("ID_1" OR "ID_2" OR ... "ID_{748}")$. If the user ID attribute satisfies any one of them, the user's ID attribute satisfies the right subtree. When the user's attributes satisfy the left and right subtrees successively, the user can be granted special permissions. If the user attribute attribute is not satisfied, it is not necessary to judge the ID attribute, determine that the user does not satisfy the access structure, cannot access the data resource, and does not assign special rights to the user. If the user already has the special right, the user is revoked special permission. If the user feature attribute is satisfied T_A but not satisfied T_ID, the user still cannot access the data resource and cannot assign the user special permission. Because the DO controls the final step of decryption, the data owner DO can control their data resources.

The access tree constructed using the ATSP-ABE algorithm is shown in Figure 9. In the access structure of the CP-ABE algorithm, T_A is improved to T_L, and T_ID

Table 2: The taxonomy veracity of the node i on the pruned and trained set

Node	N_0	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8
$m(i)$	73.81%	87.95%	55.47%	85.37%	90.48%	64.62%	76.19%	55.88%	87.10%
$m(i)'$	77.75%	90.23%	66.53%	100.00%	83.06%	60.00%	73.68%	75.47%	51.39%

Table 3: The Performance of the candidate subtree

Candidate Subtree	Leaf Node	$m(T')$	$r(T')$	$f(T')$	$P(T')$
T'_m	$N_3N_4N_6N_7N_8$	81.61%	85.51%	100.00%	85.20%
T'_{N_5}	$N_3N_4N_5N_6$	80.25%	91.26%	90.00%	86.18%
T'_{N_1}	$N_1N_6N_7N_8$	65.29%	90.52%	90.00%	79.11%
$T'_{N_5N_1}$	$N_1N_5N_6$	80.28%	96.31%	80.00%	87.47%
$T'_{N_5N_2}$	$N_2N_3N_4$	73.80%	86.31%	80.00%	80.05%
$T'_{N_5N_2N_1}$	N_1N_2	73.80%	91.32%	70.00%	81.30%
$T'_{N_5N_2N_1N_0}$	N_0	73.81%	94.93%	0.00%	75.93%

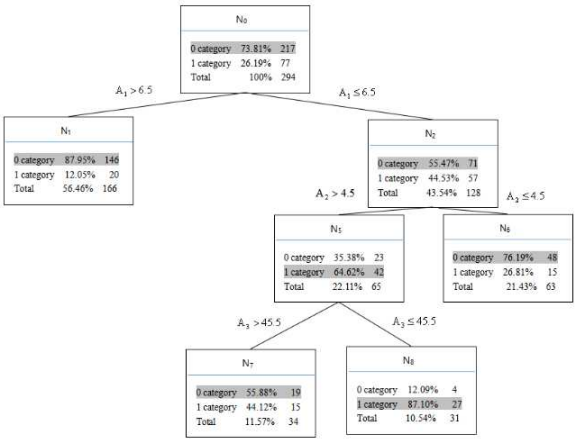

Figure 6: The REP pruned method

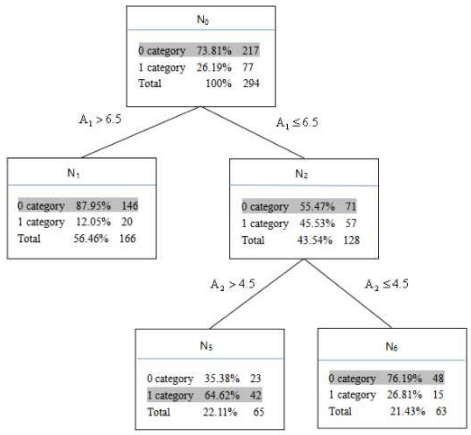

Figure 7: The pruned performance method

is improved to T_R. In the left subtree of the CP-ABE access structure. The decision tree is used to classify users, which improves the complexity of managing user attributes one by one. The pruning performance is proposed, and some feature attributes in the access tree are pruned to make the access tree The structure is more concise. Finally, the access tree structure prunes the two feature attributes $A_3 \leq 45.5$ and reducing the number of leaf nodes accessing the left subtree. Accessing the right subtree prune all of the user's 748 ID attribute nodes, replacing them with the permission access attribute node. The data owner controls the final step of decrypting the data, therefore, the access tree structure is made more concisely without affecting the performance of the algorithm, and reduced the computational complexity of the algorithm.

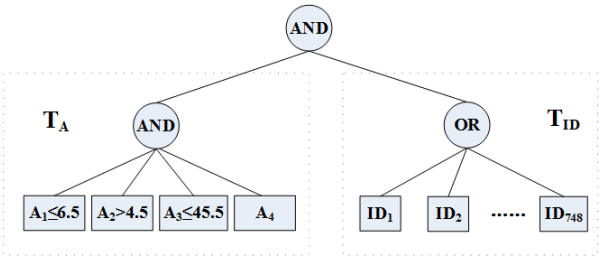

Figure 8: Transfusion user data set of CP-ABE access tree structure

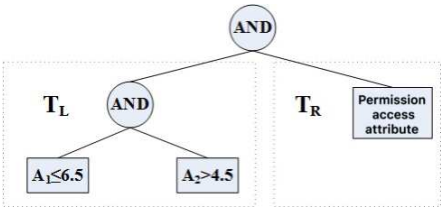

Figure 9: Transfusion user data set of ATSP-ABE access tree structure

Table 4: Classification accuracy of decision trees after pruning

		Correct quantity	Number of errors	Accuracy(%)
Pruning	Category 0	265	23	91.76
performance	Category 1	42	92	31.35
method	total	307	115	72.75
REP	Category 0	284	4	98.60
Pruning	Category 1	27	107	20.14
Method	total	311	111	73.69

6.2 Performance and Experimental Analysis

The performance of the ATSP-ABE scheme and the two typical ABE attribute schemes are compared in terms of system settings, private key generation, ciphertext size, attribute revocation, encryption and decryption, as shown in Table 5. Where n represents the number of system attributes, $L*$ represents the bit length of the element in $*$, A_u represents the number of attributes associated with the user, A_c represents the number of attributes associated with the ciphertext, A_{ID} represents the number of attributes associated with the user ID, and N_u represents the number of attributes. The number m is the size of the decryption key. The next step is to analyze the setup phase, user private key generation phase, ciphertext size, encryption and decryption algorithms phase.

Setup phase. DO defines the underlying bilinear map and generates the master key MK and the public key PK. The computational overhead of the ATSP-ABE system setup is less than the n-order multiplication of the bilinear group G_1, generating a master key MK requires a power operation. Generating a public key PK requires a power operation and a bilinear pair operation. The attribute authorization center AAC and the data owner DO each need to generate a pair of master key and public key, so the system setting phase requires a total of two power operations and two bilinear pair operations, wherein the bilinear pair is undoubtedly consuming the most time. The calculation overhead of the system setting of the ATSP-ABE scheme is the same as that proposed by Yang *et al.* [16].

Private key generation phase. The private key generation includes two parts: the feature attribute private key generation and the permission access attribute private key generation. The computational cost of this operation is the $2(A_u + 1)$-order multiplication of bilinear groups G_1. Generating a private key requires four power operations, one hash operation and one multiplication operation. The private key SK needs to be generated separately for the feature attribute and the permission access attribute, so the private key generation requires a total of eight power operations, two hash operations and two Submultiplication operation. The impact of the access tree structure of the ATSP-ABE scheme on the private key generation is to access the leaf node pruning part in the left subtree and the part that accesses the user permission attribute in the right subtree to generate the private key. Through the ATSP-ABE pruning method, the access to the left subtree prunes off some of the feature attribute nodes, so the attribute attribute managed by the attribute authorization center AAC is reduced so that the user private key generated by the attribute authorization center AAC is reduced. Accessing in the right subtree D_{id} becomes D_λ, and D'_{id} becomes D'_λ, so the calculation overhead of the user license attribute generating private key portion is reduced, thereby reducing the calculation overhead of the overall private key generation.

Ciphertext size. The ciphertext consists of a access tree, a header file, and a message body. The header file for each data consists of a collection of attributes consisting of $2(A_u+1)$ elements of G_1. Generating ciphertext requires two multiplication operations, two bilinear pair operations, two hash operations, and eight power operations. Due to the pruning process of the access tree structure, C_{id} in the information body becomes C_λ, and C'_{id} becomes C'_λ which reduces the computational overhead of the process of generating the information subject in the ciphertext, and is simplified after accessing the pruning process of the left subtree structure.Reducing the computational overhead of the entire ciphertext generation process.The summary analysis can be concluded that the computational overhead of generating the ciphertext size of the ATSP-ABE scheme is reduced by $(nA_c-2A_u-1)L_{G_1}+L_{G_2}$ compared to the computational overhead of Yang *et al.* [16], and the computational overhead is reduced by compared to Water *et al.* [26].

User attribute revocation phase. User attribute revocation includes re-generation and private key of all users, as well as ciphertext update operations. Among them, the user attribute revocation requires a thirteenth power operation, three bilinear pair op-

Table 5: Analysis and comparison of ATSP-ABE and two attribute encryption ABE schemes

Mechanism	Waters *et al.* [26]	Yang *et al.* [16]	ATSP-ABE
Structure	Linearity	Tree	Pruned tree
Complexity hypothesis	Group model	DBDH	DBDH
Revocation category	System attribute revocation, User revocation, User attribute revocation		
System settings	$3L_{G_1} + L_{G_2}$	nL_{G_1}	nL_{G_1}
Private key generation	$(1 + n + A_u)L_{G_1}$	$2(A_u + A_{ID})L_{G_1}$	$2(A_u + 1)L_{G_1}$
Ciphertext size	$(1 + nA_c)L_{G_1} + L_{G_2}$	$2(A_u + A_{ID})L_{G_1}$	$2(A_u + 1)L_{G_1}$
Attribute revocation	$(1 + 3nA_c)L_{G_1} + 2L_{G_2}$	$2N_u(A_u + A_{ID})L_{G_1} + nL_{G_2}$	$2N_u(A_u + 1)L_{G_1} + nL_{G_2}$
Encryption	$(1 + 3nA_c)L_{G_1} + 2L_{G_2}$	$(n + m + 1)L_{G_1} + 2L_{G_2}$	$(n + 1)L_{G_1} + 2L_{G_2}$
Decryption	$(1 + n + A_c)L_e + (3A_c - 1)L_{G_1} + 3L_{G_2}$	$2L_e + (m + 1)L_{G_1} + 2L_{G_2}$	$2L_e + 2L_{G_2}$
Advantage	The proxy re-encryption technology.	The fine-grained access control.	The DO can fully control the shared data, achieve the fine-grained access control, and owns high DO management attributes.
Disadvantage	No proof of the security under standard complexity assumptions, the third party and the AAC need to remain online.	The third party and the AAC need to remain online, the DO management attributes are not efficient.	The third party and the AAC need to remain online.

erations, three hash operations and three multiplication operations. Due to the ATSP-ABE pruning process on the access tree structure, the computational overhead of the system setup phase, the private key generation phase and the ciphertext size phase is reduced, so the computational overhead in the user property revocation process is also inevitable. The analysis shows that the computational overhead of the user attribute revocation of the ATSP-ABE scheme is reduced by $2N_uA_{ID}L_{G_1}$ compared to the computational overhead of Water *et al.* [26].

Encryption and decryption phase. When the information M needs to be encrypted under the condition of accessing the tree, the ciphertext CT is decrypted. The decryption operation includes two operations of accessing the left subtree and accessing the right subtree.The decryption algorithm for accessing the left subtree requires five bilinear pair operations, three multiplication operations, and two power operations. The decryption algorithm for accessing the right subtree requires two bilinear pair operations, five power operations, one multiplication operation, and two hash operations. Therefore, the total decryption algorithm requires a total of seven bilinear pair operations, four multiplication operations, seven power operations, and two hash operations. The ATSP-ABE scheme changes the access tree structure by pruning the access tree, reduces the complexity of the tree access structure, and reduces the computational overhead of accessing the right subtree encryption and decryption process. The ID-based key creation time is approximately 14-18ms. Accessing the right subtree during the encryption and decryption process eliminates the creation time of the user ID attribute, thereby reducing the computational overhead of encryption and decryption. Performing the comprehensive coefficient pruning method for accessing the left subtree reduces the subtree leaf node set W, reduces the size of the ciphertext generated during the encryption process, and reduces the computational overhead when decrypting the ciphertext.The computational overhead of the encryption operation of the ATSP-ABE scheme is reduced by $n(3A_c - 1)L_{G_1}$ compared to the scheme proposed by Waters *et al.* [16], and the scheme proposed by Yang *et al.* [26] is reduced by mL_{G_1}. Similarly, the computational overhead of the decryption operation of the ATSP-ABE scheme is reduced by $(n + A_c - 1)L_e + (3A_c - m - 2)L_{G_1} + L_{G_2}$ compared to the scheme proposed by Waters *et al.* [16] which reduces the compared to the scheme proposed by Yang *et al.* [26], compared to the two classic ABE schemes, ATSP-ABE scheme takes less time to execute, reducing the computational overhead of $2A_{ID}L_{G_1}$ size in

total, which is quite feasible for practical implementation.

This paper compares the three access structures of linear tree and pruning trees. Compare the ATSP-ABE algorithm with two typical ABE algorithms at the same security level and the same environment, and give quantitative conclusions. As shown in the experimental results in Table 6, when $n = 3$, our decryption algorithm execution time is one-fifth of Waters *et al.* [16], which is one-half of the text Yang *et al.* [26].The complexity of the access tree structure largely affects the computational overhead in the ABE algorithm. The encryption and decryption time of the algorithm depends to a large extent on the specific access structure and the set of attributes involved.

7 Conclusion

This paper proposes an access control scheme ASTP-ABE for cloud computing, which is based on CP-ABE to reduce access control policy access tree structure. Access the right subtree to prun the branch of the user ID attribute and design the permission access attribute to replace this branch with a leaf node. Accessing the left subtree generates a decision tree through the data of the user feature attribute. The algorithm selects the optimal pruning subtree as the result of pruning, and finally prunes the feature attributes in the left subtree, which are simplified in the decision tree. Simplify the pruned access tree structure and improve the efficiency of DO and AAC management and control attributes in the access policy.
In the environment of cloud computing, our solution can achieve complete control of shared data, with the help of permission access properties, DO can fully manage their data resources. Secondly, DO still controls the last step of decrypting ciphertext. In the cloud computing scenario, it can ensure that user key abuse is prevented.To solve this problem, DO can immediately terminate any user sharing data by means of user private key separation. Nextly, untrusted third-party DO can entrust semi-trusted organization to complete revocation task. DO's personal information is based on proxy re-encrypted method, which is transparent to the transferred organization. Finally, under the assumption of DBDH, our scheme satisfifies the indistinguishability of messages and ensures the confifidentiality of data in cloud computing environment.

Acknowledgments

This work was supported in part by the Key Project Foundation of Tianjin under Grant 15ZXHLGX003901, Tianjin Natural Science Foundation under Grant 19JCYBJC15800 and National Natural Science Foundation of China under Grant 61702366.

References

[1] Y. Baseri, A. S. Hafid, and S. Cherkaoui, "Privacy preserving fine-grained location-based access control for mobile cloud," *Computer and Security Journal*, vol. 73, pp. 249-265, 2018.

[2] Y. L. Chen and L. Y. Zhang, "CP-ABE based searchable encryption with attribute revocation," *Journal of Chongqing University of Posts and Telecommunication (Natural Science Edition)*, vol. 28, no. 4, pp. 545–554, 2016.

[3] X. F. Chen, Y. H. Zhang and J. Li, "Attribute-based data sharing with flexible and direct revocation in cloud computing," *KSII Transactions on Internet and Information Systems*, vol. 8, no. 11, pp. 4028–4049, 2014.

[4] T. Elomaa and M. Kaariainen, "An analysis of reduced error pruning,"*Journal of Artificial Intelligence Research*, vol. 15, pp. 163–187, 2001.

[5] J. J. A. Fournier, N. E. Mrabet and L. Goubin, "A survey of fault attacks in pairing based cryptography," *Cryptography and Communications-Discrete-Structures Boolean Functions and Sequences*, vol. 7, no. 1, pp. 185–205, 2015.

[6] J. Han, A. S. M. Kayes and A. Colman, "Ontcaac: An ontology-based approach to context-aware access control for software services," *Computer Journal*, vol. 58, no. 11, pp. 3000–3004, 2015.

[7] H. S. Hong and Z. X. Sun, "A key-insulated ciphertext policy attribute based signcryption for mobile networks," *Wireless Personal Communications*, vol. 95, no. 2, pp. 1215–1228, 2017.

[8] W. F. Hsien, C. C. Yang and M. S. Hwang, "A survey of public auditing for secure data storage in cloud computing," *International Journal of Network Security*, vol. 18, no. 1, pp. 133-142, 2016.

[9] S. T. Hsu, C. C. Yang, and M. S. Hwang, "A study of public key encryption with keyword search", *International Journal of Network Security*, vol. 15, no. 2, pp. 71–79, Mar. 2013.

[10] M. S. Hwang, W. Y. Chao, C. Y. Tsai, "An improved key-management scheme for hierarchical access control," *International Journal of Network Security*, vol. 19, no. 4, pp. 639–643, 2017.

[11] G. Karatas and A. Akbulut, "Survey on access control mechanisms in cloud computing," *Journal of Cyber Security and Mobility*, vol. 7, pp. 1–36, 2018.

[12] S. S. Keerthi K. R. K. Murthy and M. N. Murty, "Rule prepending and post-pruning approach to incremental learning of decision lists," *Pattern Recognition*, vol. 34, no. 8, pp. 1697–1699, 2001.

[13] M. C. Li, Z. Z. Guo and W. F. Sun, "Attribute based encryption with attribute-sets and multi-authority," *Journal of Chinese Computer Systems*, vol. 32, no. 12, pp. 2419–2423, 2011.

[14] Y. F. Li, W. C. Zhang and P. Wang, "Research of post-pruning decision tree algorithm based on bayesian theory in discipline evaluation," *Computer*

Table 6: Comparison of operating time between ATSP-ABE and two ABE schemes when n=3

Scheme	Encryption time	Decryption time
Water *et al.*[15]	33ms	79ms
Yang *et al.*[16]	29ms	32ms
ATSP-ABE	16ms	16ms

Engineering and Design, vol. 34, no. 11, pp. 3873–3877, 2013.

[15] Z. Liu, Z. Jiang, X. Wang, S. Yiu, R. Zhang, and Y. Wu, "A temporal and spatial constrained attribute-based access control scheme for cloud storage," *TrustCom*, pp. 614–623, 2018.

[16] F. Liu, M. Yang and J. L. Han, "An efficient attribute based encryption scheme with revocation for outsourced data sharing control," in *Proceeding of the First International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC '11)*, pp. 20–23, 2011.

[17] M. Y. Luo and J. Ling, "A security cloud storage scheme based on cipertext policy attribute-based encryption," *Journal of Guangdong University of Technology*, vol. 31, no. 4, pp. 36–40, 2014.

[18] J. F. Ma, Q. Li and R. Li, "Large universe decentralized key-policy attribute-based encryption," *Securitys and Communication Networks*, vol. 8, no. 3, pp. 501–509, 2015.

[19] C. Mao, L. Liu, Z. Cao, "A note on one outsourcing scheme for big data access control in cloud," *International Journal of Electronics and Information Engineering*, vol. 9, no. 1, pp. 29–35, 2018.

[20] K. M. Osei-Bryson, "Post-pruning in decision tree induction using multiple performance measures," *Computers Operations Research*, vol. 34, no. 11, pp. 3331–3345, 2007.

[21] T. Peng, H. Ma and Z. H. Liu, "Directly revocable and verifiable key-policy attribute-based encryption for large universe," *International Journal of Network Security*, vol. 19, no. 2, pp. 272–284, 2017.

[22] B. D. Qin, *et al.*, X. F. Huang, Q. Tao, "Multi-authority attribute based encryption scheme with revocation," in *IEEE 24th International Conference on Computer Communication and Networks (ICCCN'15)*, 2015.

[23] S. Ryu, K. R. B. Butler and P. Traynor, "Leveraging identity-based cryptography for node ID assignment in structured P2P systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 12, pp. 1803–1815, 2009.

[24] D. S. Simoes and Z. S. Donizetti, "An access control mechanism to ensure privacy in named data networking using attribute-based encryption with immediate revocation of privileges," in *Proceedings of the 12th Annual IEEE Consumer Communications and Networking Conference (CCNC'15)*, 2015.

[25] L. Touati and Y. Challal, "Efficient CP-ABE attribute/key management for Iot applications," in *Proceedings of the IEEE International Conferences on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM)*, 2015.

[26] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," *Public Key Cryptography*, vol. 6571, pp. 53–70, 2011.

[27] T. Welzer, M. Holbl and B. Brumen, "An improved two-party identity-based authenticated key agreement protocol using pairings," *Journal of Computer and System Sciences*, vol. 78, no. 1, pp. 142–150, 2012.

[28] Y. Xie, H. Wen, B. Wu, Y. Jiang, and J. Meng, "A modified hierarchical attribute-based encryption access control method for mobile cloud computing," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 383–391, 2019.

[29] C. X. Xu, A. P. Xiong and Q. X. Gan, "A CP-ABE scheme with system attributes revocation in cloud storage," in *Proceedings of the 11th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP'14)*, 2014.

[30] W. C. Zhang, and Y. F. Li, "A post-pruning decision tree algorithm based on bayesian," in *Proceeding of the International Conference on Computational and Information Sciences (ICCIS'13)*, 2013.

[31] Y. Q. Zhang, Q. Q. Zhao and G. H. Zhang, "Ciphertext-policy attribute based encryption supporting any monotone access structures without escrow," *Chinese Journal of Electronicse*, vol. 26, no. 3, pp. 640–1815, 2017.

[32] Z. Y. Zhao and J. H. Wang, "Verifiable outsourced ciphertext-policy attribute-based encryption for mobile cloud computing," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 6, pp. 3254–3272, 2017.

Biography

Ze Wang received the BE degree and the ME degree both from Xi'an Jiaotong University, China, in 1998 and 2001 respectively, and the PhD degree from Northeastern University, China, in 2004. He has been an associate professor in the School of Computer Science and Technology Tianjin Polytechnic University in China since November

2006. His primary research interests include network security, mobile computing and distributed systems.

Minghua Gao is currently a master of software engineering from Tianjin Polytechnic University. he received the bachelor degree from Tianjin University of Technology in 2018. His research interests include network security and mobile computing.

Lu Cheng is currently a master candidate from Tianjin Polytechnic University. She received the bachelor degree from Tianjin University of Technology in 2015. Her research interests include network security and mobile computing.

Shimin Sun received the BE degree and the ME degree from Chongqing University of Posts and Telecommunications and Konkuk University, in 2002 and 2007 respectively, and the PhD degree from Konkuk University, Korea , in 2012. He has been an teacher in the School of Computer Science and Technology Tianjin Polytechnic University in China since 2016. His primary research interests include network security, Wireless Network Technology and Qos algorithm optimization.