An Attribute-based Access Control with Flexible Attribute Change in Open Systems

Tao Ye [1, 2, a], Yongquan Cai [1, b]

[1]College of Computer Science and Technology, Beijing University of Technology, Beijing 100124, China;

[2]College of Computer, Qinghai University for Nationalities, Xining 81007, China.

[a]yt3262@126.com, [b]cyq@ bjut. edu. cn

Keywords: Access Control, Policies, Attribute Revocation, CP-ABE.

Abstract. The confidentiality and usability of sensitive data in outsourcing processes are particularly important in open systems. When the user's attribute is revoked or restored in the CP-ABE system, the flexibility of the dynamic change of the corresponding strategy is a challenge. In the paper, we presents a CP-ABE access control scheme which supports the dynamic changes of policy attribute. We expanded its Access Structure Tree, established a full policy access control mechanism that supports the minimum shared re-encryption key attribute set, and realized the change of the flexible access policy in the scheme.

1. Introduction

With the rapid development and extensive application of open network technology such as cloud computing, internet of things and mobile wireless networks, the network has become an open platform for a large number of cross-domain users, sharing and many-to-many relationships. Confidentiality and availability are particularly important in outsourcing process of sensitive data.

ABE comes in two flavors called key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE) [1, 2]. CP-ABE [3] is more appropriate to the data outsourcing architecture than KP-ABE because it enables data owners to choose an access structure on attributes, and to encrypt data to be outsourced under the access structure via encrypting with the corresponding public attributes.

Data owners need to start a fine-grained revocation program, and timely revoke the user's access authority, when user no longer has the right to use the data in the process of data outsourcing. So flexibility of dynamic changes for policy attribute that have been revoked or restored is an important challenge today.

There are many researches on attribute revocation in attribute-based ciphertext policy access control scheme [1-6]. In the literature [3], the 'AND' and 'OR' doors are used to realize the numerical comparison method under the inspiration of the IBE attribute revocation mechanism. Pirretti et al. proposed that the user private key is regularly updated by a trusted third party through a maintaining list of revoked attributes [5]. Hur proposed an attribute-based access control with efficient revocation (AB-ACER） that supports a fine-grained attribute revocation mechanisms [6].

Based on the above scheme, we proposes a new CP-ABE access control scheme that supports the dynamic changes of policy attribute.

2. BACKGROUND

2.1 DataStorage Architecture

As shown in Fig. 1, the scheme consists of a data storage service provider, a data owner, a trusted third party, and an ordinary user.

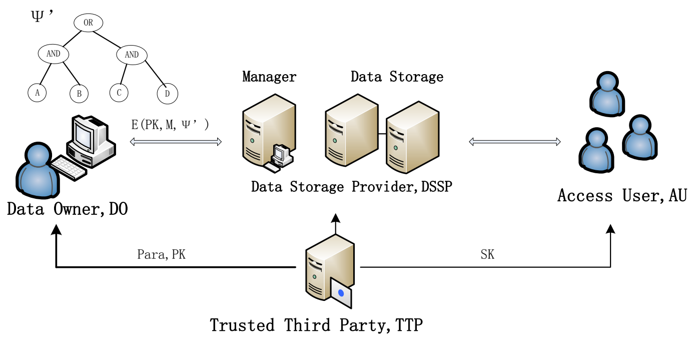

Fig. 1 Data Storage Architecture

Data Owner (DO). It is a client who owns the original plaintext data. DO is responsible to define attribute-based access policy, and to encrypts it and the original plaintext data, finally stores the ciphertext to the storage server provided by the DSSP.

Data Storage Service Provider (DSSP). It is an entity that provides data storage services for data owners and users. It is assumed that the DSSP can't be completely trusted, that is, he will perform the algorithm tasks honestly, but would also try to obtain confidential information including access strategy and plaintext.

Trusted Third Party (TTP). It is mainly used to generate the initial public parameters and MK of the system. TTP distributes public parameters and PK of the system for each data owner, and distributes, revokes and updates its private key SK for each AU. The user will send a request to the TTP, when the user or user's attribute changes. And then notify the DSSP by the TTP. In this scheme, TTP is assumed that it is a fully trusted participant in the data storage system model.

Access User (AU). It is an entity who can read the ciphertext on the data storage server. A user is able to decrypt the plaintext if and only if he has a set of attributes that satisfy the ciphertext access policy.

2.2 Definitions

Definition 1 (Access Structure). Let $P=\{P_1,P_2,\cdots,P_n\}$ be a set of parties. A collection $\Psi \subseteq 2^{\{P_1,P_2,\cdots,P_n\}}$ is monotone if $\forall B,C$: if $B\in\Psi$ and $B\subseteq C$ then $C\in\Psi$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) Ψ of nonempty subsets of $P=\{P_1,P_2,\cdots,P_n\}$, i. e. , $\Psi \subseteq 2^{\{P_1,P_2,\cdots,P_n\}}\setminus\{\phi\}$. The sets in Ψ are called the authorized sets, and the sets not in Ψ are called the unauthorized sets.

Definition 2 Let $\mathbb{U}=\{u_1,u_2,\cdots,u_n\}$ be the universe of users. n is the total number of users in the system.

Definition3 Let $L=\{\lambda_1,\lambda_2,\cdots,\lambda_q\}$ be the universe of all descriptive attributes in the system.

Definition 4 Let $G=\{G_1,G_2,\cdots G_q\}$ and $G_i\subset\mathbb{U}$ be a set of users that hold the common attribute λ_i, which is referred to as an attribute group. G_i will be used as a user access (or revocation) list to λ_i. K_{λ_i} be the attribute group key that is shared among the nonrevoked users in $G_i\in G$.

3. Scheme Construction

3.1 Algorithm

Setup (). The algorithm is run by TTP. It is a randomized algorithm that takes no input other than the implicit security parameter. It outputs the public key PK and a master key MK.

$$\begin{aligned} &PK=\left\{g,h=g^a,e(g,g)^a\right\},\\ &MK=\left\{\beta,g^a\right\} \end{aligned} \tag{3-1}$$

Where α, β is random selection and $\alpha, \beta \in Z_q^*$, $\mathbb{G}$ is a additive group of order q and g is a generator of group $\mathbb{G}_1$. $\mathbb{G}_1$ is a multiplicative group of order q, and e is a bilinear mapping of $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.

AttrKeyGen(MK,Λ,U). The algorithm is run by TTP. it takes as input the master key MK, a set of attributes $\Lambda \subseteq L$, and a set of user indices $U \subset \mathbb{U}$. It outputs a set of private attribute keys SK for each user $u_t \subset U$ that identifies with the attributes set.

$$SK_{u_t} = \left(D = g^{(\alpha+r)/\beta}, \forall \lambda_i \in \Lambda : D_i = g^r \cdot H\left(\lambda_i\right)^{r_i}, D_i^{'} = g^{r_i}\right) \tag{3-2}$$

Where $\gamma \in Z_q^*$ is randomly selected for each user, Λ is the attribute set of user U, and $\Lambda \subseteq L$, $\gamma_i \in Z_q^*$ is randomly selected for each attribute $\lambda_i \in \Lambda$.

KEKGen(U). The key encrypting key (KEK) generation algorithm is runs by the data service manager. It takes as input a set of user indices U, and outputs KEKs for each user $u_t \subset U$, which will be used to encrypt attribute group keys K_{λ_i} for each $G_i \in G$. It Generates a header message $Hdr = \left(\forall y \in \left\{Y : E_{KEK}\left(K_{\lambda_y}\right)\right\}\right)$.

Encrypt (PK,M,Ψ'). The encryption algorithm is executed by the DO. It is a randomized algorithm that takes as input the public parameter PK, a message M, and an access structure Ψ' that is defined by DO over the universe of attributes L. It outputs a ciphertext CT such that only a user who possesses a set of attributes that satisfies the access structure will be able to decrypt the message.

$$CT = \begin{pmatrix} \Psi', \tilde{C} = M \cdot e\left(g, g\right)^{\alpha s}, C = g^{\beta s}, \forall y \in Y : C_{y^+} = g^{q_{y^+}(0)}, \\ C'_{y^+} = H\left(\lambda_{y+}\right)^{q_{y^+}(0)}, C_{y^-} = g^{q_{y^-}(0)}, C'_{y-} = H\left(\lambda_{y^-}\right)^{q_{y^-}(0)} \end{pmatrix} \tag{3-3}$$

Where Ψ' is extended through adding the positive and negative attribute on the leaf nodes. The original leaf node becomes internal node 'AND'. M is the plaintext to be encrypted, $\alpha, \beta, s \in Z_q^*$, y^+ is the positive attribute of a leaf node, and y^- is the negative attribute of a leaf node. $H\left(\lambda_{y^+}\right)$ and $H\left(\lambda_{y^-}\right)$ are the public mapping function defined in the traditional algorithm CP-ABE, λ_{y^+} and λ_{y^-} are a random element mapped on group respectively.

ReEncrypt(CT,G). The algorithm is executed by DSSP. It is a randomized algorithm that takes as input the ciphretext CT including an access structure Ψ', and a set of common attribute groups G. If the attribute groups appear in Ψ', It select a re-encryption key $k_{\lambda_y} \in Z_q^*$, to complete the ciphertext CT re-encryption for attribute group $G_y \in G$;else, returns $\perp$. Specifically, it outputs a re-encrypted ciphertext CT' such that only a user who possesses a set of attributes that satisfies the access structure and has a valid membership for each of them at the same time will be able to decrypt the message.

$$CT' = \begin{pmatrix} \Psi', \tilde{C} = M \cdot e\left(g, g\right)^{\alpha s}, C = g^{\beta s}, \forall y \in Y : C_{y^+} = g^{q_{y^+}(0)}, \\ C'_{y^+} = \left(H\left(\lambda_{y+}\right)^{q_{y^+}(0)}\right)^{k_{\lambda_y}}, C_{y^-} = g^{q_{y^-}(0)}, C'_{y-} = \left(H\left(\lambda_{y^-}\right)^{q_{y^-}(0)}\right)^{k_{\lambda_y}} \end{pmatrix} \tag{3-4}$$

Decrypt (CT',SK', KΛ). The algorithm is executed by the user u_t who needs to access the encrypted file. It takes as input the ciphertext CT' which contains an access structure Ψ', a private key SK, and a set of attribute group keys K_Λ for a set of attributes Λ. The user u_t first obtains Hdr from the DSSP and the re-encryption key K_{λ_y}, and updates its SK as SK '.

$$SK'_u = \left(D = g^{(\alpha + r)/\beta}, \forall \lambda_i \in \Lambda : D = g^r \cdot H(\lambda_i)^{r_i}, D' = \left(g^{r_i}\right)^{1/k_{\lambda_i}} \right) \tag{3-5}$$

Defines the recursive algorithm $DecryptNode(CT', SK', x)$, where x represents all leaf nodes.

$$DecryptNode(CT', SK', x) = \begin{cases} e\left(D_i, C_{y^+}\right) / e\left(D'_i, C'_{y^+}\right) = e(g,g)^{rq_{y^+}(0)}, \\ e\left(D_i, C_{y^-}\right) / e\left(D'_i, C'_{y^-}\right) = e(g,g)^{rq_{y^-}(0)}, \\ \perp \end{cases} \tag{3-6}$$

Get each virtual leaf node $e(g,g)^{rq_{y\pm}(0)}$, the use of threshold-based shared secret algorithm, and ultimately can still get $e(g,g)^{rs}$, through $M = \tilde{C} / e(C, D) / \left(e(g,g)\right)$ to restore the plaintext.

4. Changes of access policy attribute

DO needn't re-encrypts the M, after the Access Policy Attribute are revoked or recovered to introduce the extended access tree Ψ', this is completed by DSSP with the calculation of resource-rich.

When some system attributes are to be revoked, DO encrypts the extracted attribute set Ψ' with the shared key KEK and sends it to the DSSP. Then the DSSP executes the algorithm $\mathrm{Re}MoveEncrypt(CT', T)$ For $\forall y \in T$. DSSP re-encrypts the ciphertext with randomly selected $s_{\lambda_y} = Z_p^*$ for each revoked system attribute λ_y. which is constructed as follows:

$$CT' = \begin{pmatrix} \Psi', \tilde{C} = M \cdot e(g,g)^{\alpha s}, C = g^{\beta s}, \forall y \in Y : C_{y^+} = g^{q_{y^+}(0)}, \\ C' = \left(H\left(\lambda_{y^+}\right)^{q_{y^+}(0)} \right)^{k_\lambda}, C_{y^-} = g^{q_{y^-}(0)}, \\ \forall y \in T : C'_{y^-} = \left(H\left(\lambda_{y^-}\right)^{q_{y^-}(0) + s_{\lambda_y}} \right)^{k_{\lambda_y}} \end{pmatrix} \tag{3-7}$$

When DO need restore some of the system properties, DO encrypts the extracted attribute set T' with the shared key KEK and sends it to the DSSP. Then the DSSP executes the algorithm $\mathrm{Re}CoverEncrypt(CT', \Psi')$. DSSP re-encrypts the ciphertext for $\forall y \in T'$, the algorithm is constructed as follows:

$$CT' = \begin{pmatrix} \Psi', \tilde{C} = M \cdot e(g,g)^{\alpha s}, C = g^{\beta s}, \forall y \in Y : C_{y^+} = g^{q_{y^+}(0)}, \\ C' = \left(H\left(\lambda_{y^+}\right)^{q_{y^+}(0)} \right)^{k_{\lambda y}}, C_{y^-} = g^{q_{y^-}(0)}, \\ \forall y \in T' : C'_{y^-} = \left(H\left(\lambda_{y^-}\right)^{q_{y^-}(0) + s_{\lambda_y}} \right)^{k_{\lambda_y}} / \left(H\left(\lambda_{y^-}\right)^{s_{\lambda_y}} \right)^{k_{\lambda_y}} \end{pmatrix} \tag{3-8}$$

The construction of the above scheme makes the change of the access policy of the data owner more flexible, and does not need the data to do the re-encryption work.

5. Conclusion

The problem that the policy attribute revocation on CP-ABE is expensive and not flexible in the open network environment is studied. We propose a minimum shared re-encryption key attribute set access control scheme that supports the revocation of policy attributes on the basis of AB-ACVE. It not only keep the original security and fine-grained access control, but also has good flexibility and efficiency.

Acknowledgements

This research is partially funded by the National Natural Science Foundation of China (61170221). With the financial support from the special fund of the network and information security research and innovation team of Qinghai University for Nationalities in China.

References

[1]. A. Sahai and B. Waters, “Fuzzy Identity-Based Encryption,” Proc. Eurocrypt ’05, pp. 457-473, 2005.
[2]. V. Goyal, O. Pandey, A. Sahai, etal. Attribute-based encryption for fine-grained access control ofen crypted data. Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006: 89-98.
[3]. J. Bethencourt, A. Sahai, B. Waters. Ciphertext-Policy attribute-based encryption, In: Proc. of the IEEE Symposium on Security and Privacy, 2007, p321-334.
[4]. B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization [M]. Public Key Cryptography–PKC 2011. Springer Berlin Heidelberg, 2011: 53-70.
[5]. M. Pirretti, P. Traynor, P. Mc Daniel, et al. Secure attribute-based systems [C]. Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006: 99-112.
[6]. J. Hur, D. K. Noh. Attribute-based access control with efficient revocation in data outsourcing systems. Parallel and Distributed Systems, IEEE Transactions on, 2011, 22(7): p1214-1221.