Achieving Lightweight and Fine-grained Access Control in Mobile Cloud Computing

Heng He[1, 2], Chuan Tian[1, 2], Yu Jin[1, 2], Wei Xia[1, 2] and Yadan Wang[1, 2]
[1]College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, China
[2]Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System, Wuhan, China
{heheng, jinyu}@wust.edu.cn, {tianchzx, wydwust}@163.com, xiawei137hao@sina.com

Keywords: Access Control, Attribute-based Encryption, Mobile Cloud Computing.

Abstract: Mobile cloud computing has emerged as one of the most promising technologies, which can greatly enlarge the application range of mobile devices by moving data storage and processing to powerful and centralized computing platforms located in clouds. However, how to keep data security and data privacy is an important and challenging issue when users outsource their sensitive data to mobile cloud for sharing. In this paper, we present a novel lightweight and fine-grained data access control scheme based on attribute-based encryption for mobile cloud computing. The analysis shows that the proposed scheme can meet the security requirement of mobile cloud computing, and provide flexible and expressive data access control policy. Furthermore, it has less computation cost compared with other schemes.

1 INTRODUCTION

With the fast development of wireless network and cloud computing (Mell and Grance, 2011) technology, mobile cloud has become the primary computing platform for many users because of the widely used mobile devices. However, security issues have been the top concern when users store sensitive data (such as photos, videos and app data) in the mobile cloud. Maintaining confidentiality and privacy of the data becomes a challenging task (Chen and Zhao, 2012). Since cloud servers are operated by commercial providers who are usually outside of the trusted domain of users, they are not entitled to access the confidential data. Furthermore, there are cases in which users (i.e. data owners) publish data in the cloud for sharing and need fine-grained data access control in terms of which users (i.e. data consumers) have the access privilege to which types of data (Yu et al., 2010).

Data access control has been well studied and various techniques have been developed to specify differential access rights for individual users. Traditional access control schemes assume that the servers are fully trusted by data owners and let the servers enforce all access control policies (Kallahalla et al., 2003), (Di Vimercati et al., 2007). However, this assumption no longer holds in cloud computing. To achieve access control of data stored on untrusted cloud servers, a feasible solution would be storing encrypted data.

A recently proposed access control model, based on Ciphertext-Policy Attribute-Based Encryption (CP-ABE), defines access control policies based on the attributes of the user (Bethencourt et al., 2007). CP-ABE is a public-key cryptography primitive that was proposed to achieve the attribute-based access control on untrusted storage. Compared to previous works, CP-ABE can achieve the fine-grainedness of data access control and meanwhile be efficient. In CP-ABE, the complexity of encryption and key management are independent from the number of system users, and is just related to the number of system attributes. There are some related schemes which exploit CP-ABE to enforce fine-grained data access control in cloud systems (Hur and Noh, 2011), (Ibraimi et al., 2009), (Wang et al., 2010). However, most of these schemes are designed for traditional cloud computing and they cannot be directly applied in mobile cloud computing, which will bring heavy computation overheads to mobile devices.

In this paper, we propose a secure and lightweight data access control scheme for mobile cloud computing. In the following of the paper we refer to our scheme as LF-ABE. By greatly reducing the computation overheads in the process of encryption and decryption, LF-ABE can achieve

He H., Jin Y., Tian C., Wang Y. and Xia W.
Achieving Lightweight and Fine-grained Access Control in Mobile Cloud Computing.
DOI: 10.5220/0006029605100515
In *Proceedings of the Information Science and Management Engineering III (ISME 2015)*, pages 510-515
ISBN: 978-989-758-163-2

Achieving Lightweight and Fine-grained Access Control in Mobile Cloud Computing

high efficiency of the system compared with the previous related schemes. Meanwhile, it provides flexible and expressive data access control policy. Furthermore, most of the computing cost are outsourced to the cloud, which can greatly decrease the computing overhead on mobile devices.

2 PRELIMINARIES

2.1 CP-ABE

CP-ABE is proposed by Bethencourt, Sahai and Waters. It is a variant of Attribute-Based Encryption (ABE) (Sahai and Waters, 2005). In CP-ABE, each user is identified with a set of attributes, and users need to register with the Trust Authority (TA), who is responsible for publishing the universe attributes and generating the related keys (including public key, master key and users' secret keys). An attribute can be any descriptive string that defines, classifies, or annotates the user, to which it is assigned. Each user's Secret Key (SK) is associated with his attributes. When Data Owner (DO) encrypts a message, he specifies an associated access policy tree over attributes, which is embedded into the ciphertext in the process of encryption. The access tree is used to descript an access policy that specifies which combination of attributes can decrypt the ciphertext. Let tree T whose root node is r represent the access policy. Each inner node of T is a logic operator, such as "AND", "OR" and "OF", while each leaf node represents an attribute. According to the idea of secret sharing, each node of the access tree represents a secret. In the encryption phase, we need to top-down recursively assign a secret to each node. While in the decryption phase, we need to bottom-up recover the secret of the root node. A user will only be able to decrypt a ciphertext if that user's attributes pass through the ciphertext's access tree.

2.2 Bilinear Map

Let $G_0$and G_1 be two multiplicative groups of prime order p, Q is a generator of G_0. Map e: $G_0 \times G_0 \rightarrow G_1$ is a bilinear map with the following properties:

Bilinear: for all a, b $\in Z_p^*$ and R, $S \in G_0$, we have $e(R^a, S^b) = e(R, S)^{ab}$.

Non-degenerate: $e(Q, Q) \neq 1$.

Computable: the group operation in G_0 and the bilinear map e are both efficiently computable.

In addition, the map e is symmetric:

$$e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a).$$

2.3 Shamir's Secret Sharing Scheme

In Shamir's secret sharing technique (Shamir, 1979), a secret s is divided into n shares in such a way that any subset of t shares, where $t \leq n$, can together reconstruct the secret; no subset smaller than t can reconstruct the secret. The technique is based on polynomial interpolation where a polynomial $y = f(x)$ of degree t-1 is uniquely defined by t points (x_i, y_i). The details of the scheme are as follows:

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

3 LF-ABE

3.1 Scheme Overview

As it is known, due to the differences in hardware and platforms, computing capabilities of the mobile device is far weaker than the desktop PC. It is difficult for lightweight mobile devices to finish the encryption and decryption operations of CP-ABE on their own. To reduce the computation overheads at mobile client, we take consideration of securely outsourcing computation intensive CP-ABE operations to Encryption Cloud Service (ECS) and Decryption Cloud Service (DCS) located in cloud. Figure 1 gives the framework of LF-ABE.

We construct our LF-ABE scheme based on Ibraimi's CP-ABE algorithm (EPS-ABE) (Ibraimi et al., 2009) since it is very secure and efficient. The encryption complexity grows linearly on the size of

511

access policy. For the purpose of outsourcing the computation cost at mobile client, we define a virtual attribute which is owned by every user. We denote the virtual attribute as Att_{Vir}. The access tree T is constructed by T_{Pol} and T_{Vir}, and they are connected with an "AND" operator, that is $T = T_{Pol} \wedge T_{Vir}$. T_{Pol} is the real access policy, while T_{Vir} only contains Att_{Vir}.

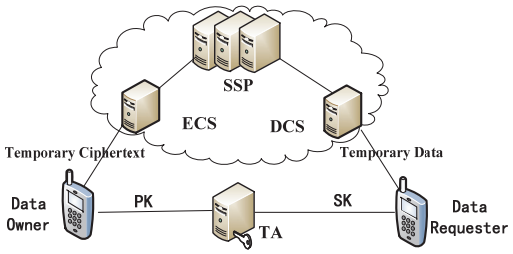

Figure 1: The framework of LF-ABE.

First, we utilize Shamir's secret sharing technique to assign each leaf node of the access tree a secret s_i. Specially, denote the secret of Att_{Vir} as s_{Vir}. Then, we use each leaf node's secret to perform a point multiplication operation which is used to encrypt the file. The number of point multiplications grows linearly on the number of the leaf nodes. Apparently, the leaf nodes in T_{Vir} are far less than those in T_{Pol}, so the point multiplication operations in are far less than T_{Vir}. To outsource the computation cost of mobile client, DO sends each s_i in T_{Pol} to ECS, then ECS performs the point multiplication operations and returns a temporary ciphertext, while DO only needs to make use of s_{Vir} to perform point multiplication operation once at mobile client.

In the decryption phase, the bilinear pairing operations over ciphertext and private key take heavy overhead. Denote every secret key component as D_i and Data Requester (DR) sends each D_i except D_{Vir} to DCS. Then, DCS runs the decryption algorithm to compute a temporary data, and returns it to DR. Finally, DR takes advantage of the temporary data and D_{Vir} to recover the real file locally.

3.2 The Construction of LF-ABE

LF-ABE consists of four algorithms. The detailed construction of LF-ABE is as follows:

1) ***Setup* (*k*).** TA runs the setup algorithm to generate a system Public Key (*PK*) and a system Master Key (*MK*). The input is a security parameter k which determines the size of the groups. This algorithm generates an elliptic curve group G_0 of prime order p with a generator Q, which is a point on the elliptic curve. The bilinear map is e: $G_0 \times G_0 \rightarrow G_1$. Then, generate the universe attribute set $\Omega = \{a_1, a_2, \ldots, a_n\}$. Note that each user's attribute set contains a virtual attribute. Next, it chooses a random elemen $t_j \in Z_p$for each attribute a_j in Ω. Then, it selects the last random element $\alpha \in Z_p$. Finally, it generates *PK* and *MK* as follows:

$$PK = (G_0, Q, e, y = e(Q, Q)^{\alpha}, T_j = t_j Q (1 \le j \le n))$$

$$MK = (\alpha, t_j (1 \le j \le n))$$

Every party can get access to *PK* while *MK* is secretly held by TA and not published.

2) ***KeyGen* (ω, *MK*).** TA runs the key generation algorithm, which takes as input the master key *MK* and a set of attributes that describe the identity of the user. It outputs a private key *SK*. This algorithm first chooses a random $r \in Z_p$ and computes $D_0 = (\alpha - r)Q$. Then for each attribute a_j in ω, compute $D_j = (rt_j^{-1})Q$. For the purpose of outsourcing the computation cost of mobile client and preserving the data privacy, a virtual attribute named Att_{Vir} is defined. Specially, denote the secret key component of virtual attribute as D_{Vir}. The user's private key *SK* is:

$$SK_{\omega} = (D_0, \forall a_j \in \omega: D_j)$$

3) ***Encrypt* (*m*, *T*, *PK*).** The encryption algorithm encrypts a file m under the access tree T. Att_{Vir} is attached to the access tree as the right sub-tree. The access tree is redefined as $T = T_{Pol} \wedge Att_{Vir}$. T_{Pol} is the real access policy that not need to be modified in any form.

The detailed encryption algorithm processes as follows:

A) Select a random element $s \in Z_p$. Set the value of the root node of T to be s, mark the root node as assigned and all child nodes as un-assigned. For each un-assigned non-leaf node, recursively do the following:

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

Achieving Lightweight and Fine-grained Access Control in Mobile Cloud Computing

node a share secret $s_i = f(i)$ is assigned. Mark this node assigned. If the symbol is "OR", and its child nodes are marked un-assigned, the secret s is divided using (t, n)Shamir's secret sharing technique where t = 1 and n is the number of child nodes. To each child node a share secret $s_i = f(i)$ is assigned. Mark this node assigned. Figure 2 describes the detailed progress of assigning secrets to each node of the access tree.

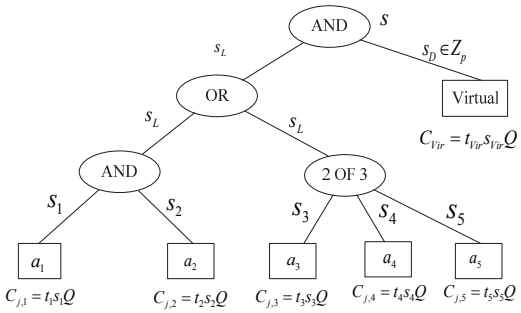

Figure 2: The way to assign secrets.

Then DO computes C_0, C_1 and C_{Vir} at local mobile client as following to get the temporary ciphertext:

$$CT_{Vir} = (C_0 = sQ,\ C_1 = me(Q, Q)^{\alpha s},\ C_{Vir} = t_{Vir} s_{Vir} Q)$$

After completing the above steps, DO sends every s_i, and $CT_{Vir}$to the encryption cloud service ECS, where $a_i \in T_{Pol}$, and i denotes the index of the attribute in the access tree. The index values are uniquely assigned to leaf nodes in the access tree in an ordering manner.

B) For each leaf attribute $a_{j,i} \in T_{Pol}$, ECS computes $C_{j,i} = s_i T_j$. By doing this, most of the computation overheads are outsourced to ECS. Then we get the other temporary ciphertext CT_{Pol}:

$$CT_{Pol} = (\forall a_{j,i} \in T: C_{j,i})$$

C) ECS uses the two temporary ciphertexts to generate the following complete ciphertext:

$$CT = CT_{Pol} \wedge CT_{Vir} = (\forall a_{j,i} \in T: C_{j,i}, C_{Vir}, C_0, C_1)$$

Finally, ECS sends CT to the Storage Service Provider (SSP) .

4) *Decrypt* (CT, SK_ω). The decryption algorithm decrypts the file m from the ciphertext using user's private key SK_ω. The decryption algorithm is computationally expensive since bilinear pairing operation over ciphertext and private key takes heavy overhead. To reduce the computation overheads of mobile client, we outsource most of the computation overheads to DCS. The decryption process is decomposed into the following three steps:

A) DR first requires data from SSP. SSP checks whether DR's attribute set satisfies the access tree T embedded in the ciphertext. If so, SSP sends CT to DCS. Meanwhile, DR sends all of the components of SK_ω except D_0 and D_{Vir} to DCS. Then DCS chooses the smallest attribute set $\omega' \in \omega$ that satisfies T. For each a_j in ω', compute:

$$\begin{aligned} A &= \prod_{a_j \in \omega'} e(C_{j,i}, D_j)^{l_i(0)} \\ &= \prod_{a_j \in \omega'} e(t_j s_j Q, rt_j^{-1} Q)^{l_i(0)} \\ &= e(Q, Q)^{rs_L} \end{aligned}$$

DCS then sends parameter A to DR.

B) DR computes locally:

$$\begin{aligned} B &= e(C_0, D_0)\, e(C_{Vir}, D_{Vir})\, A \\ &= e(sQ, (\alpha - r)Q) e(t_{Vir} s_{Vir} Q, rt_{Vir}^{-1} Q) e(Q, Q)^{rs_L} \\ &= e(Q, Q)^{\alpha s} \end{aligned}$$

C) DR performs the final step and gets the file m:

$$m = C_1 / B = me(Q, Q)^{\alpha s} / e(Q, Q)^{\alpha s}$$

4 SECURITY AND PERFOMANCE EVALUATION

4.1 Security Discussion

We now briefly discuss security properties of LF-ABE.

In the encryption algorithm, ECS is just responsible for dealing with the computations that related to access control and it does not get any direct secrets of the data content. The ciphertext component C_1 that related to the file m is computed by mobile device locally. According to Shamir's secret sharing technique, recovering the master secret requires the cooperation of all parties if the operator is "AND". That is to say, though ECS has most of the secrets of the access tree, it still cannot recover the master secret because the secret of virtual attribute is secretly held by DO. While in the decryption phase, sending all the secret key components except D_0 and D_{Vir} to DCS will also not reduce security level. DCS can use these secrets to get a temporary result, but the final setp of

513

decryption is performed by the decryptor at mobile client.

In LF-ABE, the KeyGen algorithm generates a different random value r for each user, this means each user's secret key is randomized and can not be combined with the others'. So malicious users can not use collusion to expand their access permission in LF-ABE, including ECS and DCS.

4.2 Performance Evaluation

The computation operations in ABE system are multiplication, point multiplication, exponentiation and pairing. Due to the computation overheads of processing multiplication is much less than that of the other three operations, its influence on the system performance can be neglected. So we just take consideration of pairing, point multiplication and exponentiation.

To facilitate the evaluation, we compare the computation overheads in Desktop PC, laptop PC and smartphone. The testing environment is Java platform. For the convenience of elaboration, denote ω as the set of attributes (except the virtual attribute) the user owns, ω' is the set of attributes satisfying the access tree where $\omega' \in \omega$ and T is the set of attributes in the access tree. Assume that all the schemes use the same bilinear map e: $G_0 \times G_0 \rightarrow G_1$. C_0 denotes point multiplication on elliptic curve group G_0, C_1 denotes exponentiation on G_1 and C_e denotes pairing.

Table 1: Computing time on different devices (Unit: ms).

Device	CPU	C_0	C_1	C_e
Desktop	3.4GHz	24.7	3.1	25.1
Laptop	2.0GHz	42.8	7.2	49.6
Smartphone	1.0GHz	159.0	48.7	314.7

We can see clearly from Table 1 that, the computing capabilities between different devices are very large. Consequently, running ABE system on lightweight mobile devices will take heavy overhead.

In Table 2, we compare the computation overheads between our scheme and the EPS-ABE scheme at mobile client under the assumption that their CP-ABE algorithm is directly used to mobile cloud and does not perform any outsourcing operations. The key generation algorithm is performed by TA and it has no connection with the mobile user, so we do not describe it in Table 2. Table 2 shows that compared with the EPS-ABE scheme, most of the computation operations in LF-ABE are outsourced to ECS and DCS, the computation overheads are greatly reduced at mobile client.

Figure 3 compares the computation time of encryption and decryption at mobile client between LF-ABE and EPS-ABE. The time of the two phases in EPS-ABE scheme is linear with the number of attributes in the access tree, while in LF-ABE the time is constant. Figure 3 shows that if the number of the attributes is large enough, more than 90% of the computation overheads are outsourced to the cloud services.

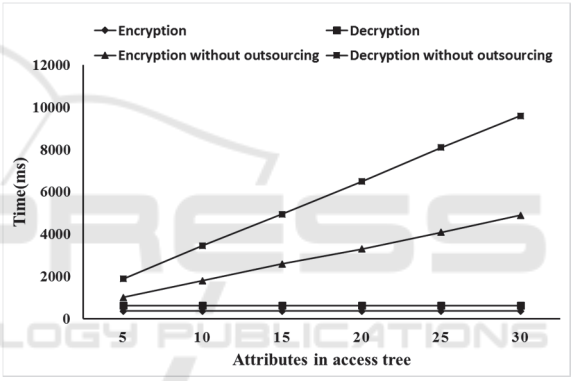

Figure 3: The comparison of computation time between LF-ABE and EPS-ABE at mobile client.

5 CONCLUSIONS

In this paper, we propose a novel lightweight and fine-grained access control scheme for mobile cloud computing named LF-ABE, which meets the requirements of data security and privacy in mobile cloud computing. Since there are no special restrictions on the access policy, LF-ABE provides expressive and flexible data access control.

Table 2: The comparison of computation overheads between LF-ABE and EPS-ABE at mobile client.

Scheme	LF-ABE			EPS-ABE		
Operation	C_0	C_1	C_e	C_0	C_1	C_e
Encryption	2	1	/	$\|\omega\|+1$	1	/
Decryption	/	/	2	/	/	$\|\omega'\|+1$

Achieving Lightweight and Fine-grained Access Control in Mobile Cloud Computing

Meanwhile, by outsourcing most of the computation overheads to the encryption and decryption cloud services, mobile devices can easily complete the data processing operations.

ACKNOWLEDGEMENTS

This work was supported by the Youth Talent Project of the Science and Technology Research Program of Hubei Provincial Education Department under Grant No. Q20151111, and Young Scientist Foundation of Wuhan University of Science and Technology under Grant No. 2013xz012, No. 2014xz019, and No.2015XG005.

REFERENCES

Mell, P., Grance, T., 2011. The NIST definition of cloud computing, *NIST Special Publication 800-145*.

Chen, D., Zhao, H., 2012. Data security and privacy protection issues in cloud computing. In *Proceedings of the 1st International Conference on Computer Science and Electronics Engineering,* pp. 647-651. IEEE Computer Society.

Yu, S., Wang, C., Ren, K., Lou, W., 2010. Achieving secure, scalable, and fine-grained data access control in cloud computing. In *Proceedings of the 29th International Conference on Computer Communications,* pp. 1-9. IEEE Computer Society.

Kallahalla, M., Riedel, E., Swaminathan, R., Wang, Q., Fu, K., 2003. Plutus: scalable secure file sharing on untrusted storage. In *Proceedings of the 2nd Conference on File and Storage Technologies,* pp. 29-42.

Di Vimercati, S. D. C., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P., 2007. Over-encryption: management of access control evolution on outsourced data. In *Proceedings of the 33rd International Conference on Very Large Data Bases,* pp. 123-134. VLDB endowment.

Bethencourt, J., Sahai, A., Waters, B., 2007. Ciphertext-policy attribute-based encryption. In *Proceedings of the 28th IEEE Symposium on Security and Privacy,* pp. 321-334. IEEE Computer Society.

Hur, J., Noh, D. K., 2011. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Transactions on Parallel and Distributed Systems,* 22(7), pp. 1214-1221. IEEE Computer Society.

Ibraimi, L., Petkovic, M., Nikova, S., Hartel, P., Jonker, W., 2009. Mediated ciphertext-policy attribute-based encryption and its application. In *Proceedings of the International Conference on Information Security Applications,* pp. 309-323. Springer Berlin Heidelberg.

Wang, G., Liu, Q., Wu, J., 2010. Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In *Proceedings of the 17th ACM Conference on Computer and Communications Security,* pp. 735-737. ACM Press.

Sahai, A., Waters, B., 2005. Fuzzy identity-based encryption. In *Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques,* pp. 457-473. Springer Berlin Heidelberg.

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

Ibraimi, L., Tang, Q., Hartel, P., Jonker, W., 2009. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In *Proceedings of the 5th International Conference on Information Security Practice and Experience,* pp. 1-12. Springer Berlin Heidelberg.

515