Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 3, March 2019, pp. 969-976
DOI: 10.23940/ijpe.19.03.p26.969976

Dynamic Access Control of Encrypted Data in Cloud Computing Environment

Shuaiqiu Xiang* and Zhenjia Zhu

Shenzhen Institute of Information Technology, Shenzhen, 518000, China

Abstract

The confidentiality of data is a difficult problem in a cloud computing environment. The DAC technique (Dynamic Access Control) based on encrypted data is an important way to solve this problem. In the current access control techniques based on encrypted data, the high-security requirement of data and frequent policy updates lead to the high cost of owner right update which seriously restricts the flexibility of access control. A DCA method based on CACDP encrypted data is proposed. The selective encryption model is built. In the model, a key derivation diagram is generated to distribute the key. In the case of ensuring the confidentiality of cloud computing access control, the key is less in the system. The proposed CACDP scheme includes the key management mechanism based on the binary Trie tree. Based on this, the ELGamal-based proxy re-encryption mechanism and double layer encryption strategy are used to transfer the partial spending of the key and data update to the cloud to ease the DO authority management burden and increase the efficiency of DO. Then, the DCA method of encrypted data in cloud computing environment is researched. Experimental results show that our proposed method can effectively improve the flexibility of encrypted data access control.

Keywords: cloud computing environment; encrypted data; dynamic access; data control

(Submitted on November 3, 2018; Revised on December 1, 2018; Accepted on January 2, 2019)

1. Introduction

Cloud computing gathers together a variety of computing infrastructures to form large scale shared virtual resources and to provide services to users through the network. The computing infrastructures can provide software service, hardware service, or data storage service. In cloud computing, the security is facing a more severe test than ever [1]. How to resist the security impact of cloud computing will become a key factor affecting the further development of cloud computing technology [2].

Access control is a way to display or restrict user access capability and scope in some way. It is an important basis for system confidentiality, integrity, availability, and legitimate usage, and is one of the key strategies for network security and resource protection. When users access resources, they must access according to their own permissions, and cannot implement access behavior beyond their own permissions. Because access control uses the principle of minimum privilege, when the user applies for permissions, the system administrator according to the characteristics of each user assigns the minimum right to complete their own task. The user will not obtain beyond the right to complete their work [3-5]. The basic goal of access control is to prevent unauthorized users from illegally accessing the protected data resources and allow the authorized users to access the protected resources reasonably. This is a necessary feature of a security system. In order to improve the security and flexibility of DCA of encrypted data, the DCA of encrypted data in the cloud computing environment is researched [6-8].

A cloud computing privilege revoking optimization mechanism based on dynamic re-encryption (DR-PRO) is proposed by Du et al. [9]. Experimental results show that DR-PRO can effectively upgrade the performance of user access revoking in cloud computing service, but the flexibility of access control is poor. The attribute encryption scheme in cloud computing

* Corresponding author.
E-mail address: xiangshuaiqiu2018@126.com

is often used to implement fine-grained access control by Wang et al. [10]. The scheme is proved to be safe under the standard model, and experimental verification and analysis are performed. Compared to the existing schemes, although additional computing load is added in order to realize access control strategy hiding and solve the key escrow problem, the flexibility of DCA of encrypted data is reduced.

2. Research on DCA Method of Encrypted Data

2.1. Model of Selective Encrypted Data

The basic model of selective encryption is as follows. The data owner encrypts the file f with the symmetric encryption algorithm and stores it on the cloud server. Then, the owner realizes the sharing and access control between the user groups through the distribution of the symmetric key, that is, when the user u has access to f, the data owner distributes the decryption key of f to the user u through the key distribution mechanism. In this paper, each shared user has only one symmetric key as its user key [11].

Access control policy is the data owner's authorization information for each shared user. The definition is as follows.

Definition 1 (Access control policy): Assume U and F are the user set and file set in system. u and f represent a user and a file, $p = \langle u, f \rangle$ denotes the authorization of cloud data owners to allow the user u to access the file f, and P is the set of authorization p. Then, the definition of the access control policy on U and F is $ACP = \langle U, F, P \rangle$.

To achieve access control of encrypted data, the key distribution is chosen in the form of a symmetric key derivation graph, which can effectively reduce the key management burden of the data owner. The main steps of selective encryption schemes to generate open information for access control are as follows.

(1) Input ACP and generate key derivation graph. The generation is as follows.

Create a vertex. The set of each shared user and access user of each file is regarded as a vertex in the key derivation graph. In the key derivation graph, the vertex representing the shared user is the user vertex, and the vertex representing the access user set of file is the file vertex. The user vertex key is the same as the user key, and the user access set of the user vertex contains only the user represented by it. The file vertex key is the same as the file decryption key. The access user set of the file vertex is the access user set of the file [12].

(2) Generate a key label. After obtaining the key derivation graph, the data owner generates a label for each key and a list of labels according to Table 1. In Table 1, l_u represents user key label of the user u as the decryption key label of the file f.

(3) Generate a token. For each directed edge in key derivation graph $e\langle v_i, v_j \rangle$, token is generated according to Table 2. Assume the keys of v_i and v_j are k_i and k_j, the labels of k_i and k_j are l_i and l_j.

After obtaining the user key label list, file decryption key label list, and token list, the data owner stores these lists as the public information in the cloud server.

Table 1. Format of label

User/File	Label of user key / Label of file decryption
u/*f*	l_u/l_f

Table 2. Format of token

Decryption label	Obtained label	Ciphertext
l_i	l_j	k_j /hash(k_j, l_j)

Example (Example of selective encryption: public information generation for access control in terms of its policy): Its policy defined by data owner is shown in Table 3. Table 4 is file access user set list obtained from Table 3.

Table 3. Example of access control policy

ACP=（U,F,P）
U={u_1, u_2, u_3, u_4, u_5, u_6}
F={f_1, f_2, f_3, f_4, f_5, f_6, f_7}
P=(u_1, f_1), (u_1, f_2), (u_2, f_1), (u_2, f_2), (u_2, f_3), (u_2, f_4), (u_2, f_5),
(u_2, f_6), (u_3, f_2), (u_3, f_2), (u_3, f_3), (u_3, f_3), (u_3, f_4), (u_3, f_5),
(u_3, f_6), (u_4, f_3), (u_4, f_4),(u_4, f_5), (u_4, f_6), (u_5, f_3), (u_5, f_4)

Table 4. File access user set

File	Access user set
f_1	(u_1, u_2)
f_2	(u_1, u_2, u_3)
f_3	(u_2, u_3, u_4, u_5)
f_4	(u_2, u_3, u_4, u_5)
f_5	(u_2, u_3, u_4)
f_6	(u_2, u_3, u_4, u_6)
f_7	(u_5, u_6)

Assume the user keys of users $u_1, u_2, u_3, u_4, u_5, u_6$ are $k_1, k_2, k_3, k_4, k_5, k_6$, the decryption keys of files $f_1, f_2, f_3, f_4, f_5, f_6$ is $f_7, f_8, f_9, f_{10}, f_{11}, f_{12}$. f_3 and f_4 have the same access user set, so they are encrypted with the key k_9. According to Table 3, the generation of key derivation graph is shown in Figure 1. $v_1 \sim v_6$ are user vertexes. $v_7 \sim v_{12}$ are file vertexes. Assume the labels of keys $k_1 \sim k_{12}$ are $l_1 \sim l_{12}$. Finally, the data owner stores encrypted data on the cloud server.

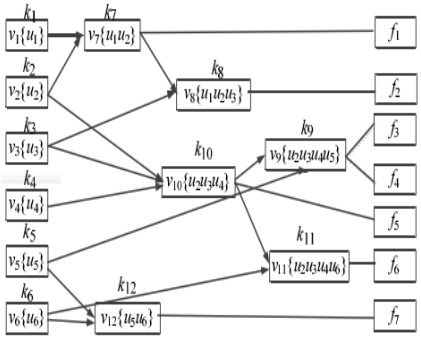

Figure 1. Key derivation graph

2.2. DCA of Encrypted Data based on CACDP Scheme

In the ParaInitial stage, the DO generation system initializes the open parameters of prime number P, domain Z, generator g, random parameter r, and $g^r \bmod P$. In FilePublish stage, the key tree is constructed by the key tree construction module, and the encryption and publishing of the file is realized [13]. In the process of the DO adding a new file to the CSP , first a token chain for requesting access control vector $ACS(f)$ with root node as starting point and path as file f for the CSP, and then the key of the DO dk and token chain $chain$ are used to calculate the key *vk* of the tail node. After obtaining the key *vk*, there are two conditions. First, if the length of $chain$ is the same as $ACS(f)$, it shows that there is a corresponding security resource class in the key tree, then vk is used to encrypt the symmetric key k_f of f. The encrypted file with signature uploads to CSP, otherwise, according to $ACS(f)$ the branch of the Keyder-Trie tree and Keyder-Trie tree node are created. The key is distributed to the node and token chain is created from top to bottom. By using encryption key k_f of the file with the asymmetric key of the root node, the Trie tree node information, token chain, data and key ciphertext are uploaded to the server. In the process of a file upload, the following 2 points need to be paid attention. (1) The character 1 represents the corresponding role has access permission of the file and needs to increase the role token of the corresponding role to the node. (2) The key of the leaf node of the key management tree is the public key pair $(g^r \bmod p, x)$. Non-leaf nodes randomly generate symmetric keys, so when creating new node, it is necessary to determine whether the node is a leaf node. See algorithm 1. The access control matrix in algorithm 1 is converted to the key management tree by the algorithm as shown in Figure 2.

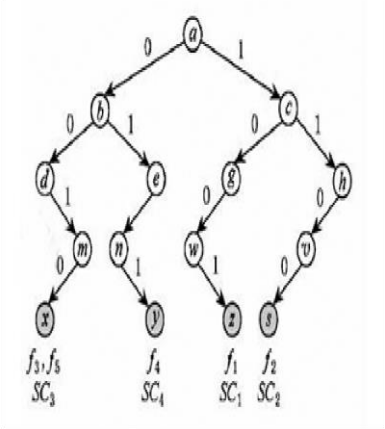

Figure 2. File distribution

In the KeyDistribution stage, according to the role of the user, the DO sends the corresponding role keys to all users through the secure channel. CSP distributes the corresponding outer key rk' to the corresponding user.

In the AccessFile stage, the file access module is used to obtain the plaintext of the file. The user sends the user role information R and access control vector $ACS(f)$ of file f to CSP. CSP returns the token chain corresponding to the file, file key ciphertext C_{kf} , file encryption parameter *encflag* , and file ciphertext. In the case of *encflag* is 0, the user resolves the token chain by holding the key, and encrypts the file key k_f , which is an asymmetric key (The private key is a , and the public key is $g^a mod\ P$). Then, the decryption algorithm is used to obtain file key k_f . Finally, the plaintext of file f is obtained. In the case of *encflag* is 1, the outer ciphertext is decrypted to obtain the ciphertext of the file, and then the key k_f of the file f is obtained to decrypt plaintext data of the file f .

In the scheme, policy change mainly includes two cases: FilePrivUpdate and RoleUpdate. The two cases are completed by the authority update module and role update module combined with the re-encryption key module and data management module, respectively.

For the case of FilePrivUpdate, file access permission update is divided into file access permission and recovery. Permissions recovery changes the value of the role R in the access control matrix of file f from 1 to 0, and the file authority is changed from 0 to 1. For the key management tree, the essence of file access permission change is the switch between different security classes for the file and file key. Therefore, when the access permission is granted, the re-encryption key $(g^a mod\ P)^{a-b}$ is generated by using the re-encryption key module. There is no need to update the file key k_f , but only the need to update the key a of k_f . Finally, the file key k_f is re-encrypted into a new key encrypted ciphertext by CSP. When revoking permission, not only is the key a of the encrypted file key k_f updated, but also the file key k_f is changed to k_f' , and finally the double layer encryption strategy is used to update the data ciphertext. See algorithm 2.

Algorithm 2: FilePrivUpdate.

Input: File index *ID* , file source and destination access control vector *sacl* , *dacl* , policy update type *type* .

Output: Updated key management tree.

(1) First obtain the token chain *schain* and *dchain* of *sacl* and *dacl* .

(2) Obtain the key of the tail node of two token chains *oldsk* and *newsk* .

(3) If $|dchain| \prec |dacl|$, go to step (4), otherwise step (5).

(4) Generate the Trie tree node and token chain of $dacl$ by using algorithm 1 and obtain the asymmetric key $newsk'$ of the leaf node corresponding to the branch i and assign to the $newsk$.

(5) If $type = 0$, it means requiring authorization, go to step (6) and (7), otherwise it means revoking, go to step (8).

(6) Re-encryption parameters k_f, $k_f'^{-1}$ and $oldsk - newsky$ are generated by re-encryption module. $rekey = \frac{k_f}{k_f'}(g^r \bmod P)^{a-b}$ is generated by CAP.

(7) If $encflag = 0$, re-encryption is implemented by CSP, and then the value of $encflag$ is changed to 1, otherwise, the plaintext of file f is obtained by file access function. The new encrypted key is used to generate a new ciphertext and is uploaded to CSP, and then the value of $encflag$ is changed to 0.

(8) Re-encryption parameter $oldsk - newsky$ is generated by the re-encryption module. $rekey = (g^r \bmod P)^{a-b}$ is generated by CSP.

Figure 3 shows the process of granting access to the file f_5 for the role R_2. The process is to transform the file k_f and its corresponding key k_f from the security resource class SC_3 to SC_4. The DO gives the old and new access control vectors $ACS(f) = 0010$ and $ACS(f)' = 0010$ of the file f_5 to CSP. The taken chain of root node and leaf node is obtained and returned to the DO. The DO calculates the security key, which is sent to the CSP according to the re-encryption key. The file key k_f is encrypted. The idea of role permission recovery is similar to the granting of permission.

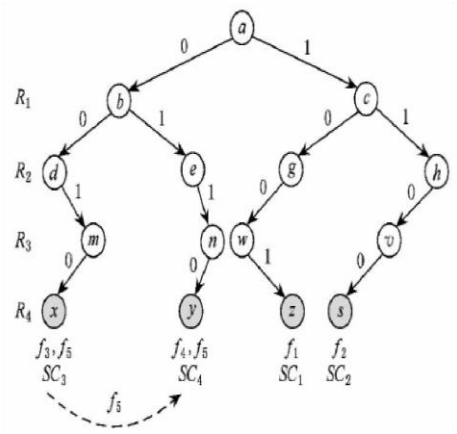

Figure 3. File permission change

For the case of RoleUpdate, the DO changes the role of the user from sr to dr. CSP returns all tokens that need to update the node key according to the request, that is, returns the intersection nodes of the node set which sr can access and the node set which dr cannot access and token chain. The token chain set consists of the three categories of token chain set with leaf nodes LT, token chain set without leaf nodes NLT, and role token set RT.

Algorithm 3: RoleUpdate

Input: Source role and destination role R_1 and R_2.

Output: Updated key management tree.

(1) If $L_{R_1} \prec L_{R_2}$, go to step (2), otherwise step (5).

(2) Obtain key node set CN of R_1 in L_{R_1}.

(3) Iterate over CN and select node n.

(4) Preorder traversal binary tree, and obtain the node set which key to be updated in L_{R_1} layer and token chain including role token (node satisfies the condition that R_1 can access but R_2 cannot), go to step (8).

(5) Obtain key node set BN which R_2 does not hold in L_{R_2} layer.

(6) Iterate over BN.

(7) Preorder traversal binary tree with root node of n, and obtain the node set which key to be updated and token chain including role token.

(8) Return the token information to be updated to the DO. The DO distributes the key to the node to be updated and form the new token chain. Update all role token of the role R_1 and distribute the new key.

(9) Generate a new symmetric key k_f' for SC_i and re-encryption key parameters of $k_f, k_f'^{-1}$, and $oldsk - newsky$.

10) If $encflag = 0$, encrypt ciphertext data by CSP and change the value of $encflag$ to 1, otherwise, obtain plaintext of the file f by file access function. Use the new encryption key $k_f^{'}$ to generate new ciphertext and upload CSP, and set $encflag$ to 0.

The DCA of encrypted data is accomplished by the above process.

3. Experimental Results and Analysis

The cloud computing environment is built by Opennebula. There are ten blade servers with Xen virtual monitor of version 3. three are at the bottom. Cloud computing providers CSP, users and data owners, are all Xen virtual machine. The operating system of CSP is 64 b Red Hat Enterprise 5. 5 with 4 VCPUs and 8GB memory. The operating system of the user virtual machine is Windows XP with 2 VCPUs and 2GB memory. Users, data owners and CSP use Gigabit switch to connect. The one-way Hash function used to generate tokens in the key management tree is SHA1. File encryption uses 128B AES encryption algorithm, and the file key encryption uses EI and the Uamal algorithm. The prime number P is 160b. In experiments, the encryption algorithm of CP-ABPRE scheme is 128B AES encryption algorithm.

Figure 4 shows the time cost of the key management tree generation of the CACDP scheme. From Figure 4, it can be seen that in the case of a certain number of security classes, the generation time of the key management tree is linearly related to the number of roles, while in the condition of the fixed role size, the spanning time of key management tree in the cases of $|SCG| = 1000$, $|SCG| = 2000$, and $|SCG| = 3000$ is increased in turn. This is because in the extreme case of CACDP scheme, the key management tree has only $|SCG|$ $|R|$ nodes for each security class, and the corresponding role token should be constructed according to the access control matrix. When the number of roles and security classes are 400 and 3000, respectively, the spanning time of the management tree is about 21s, within the acceptable range.

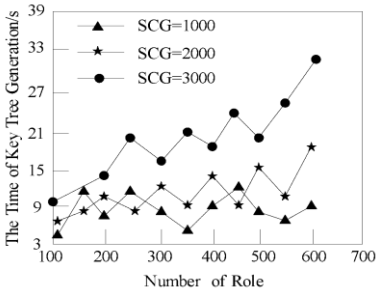

Figure 4. Time cost of the key management tree generation

In this scheme, the total length of ciphertext is given by

$$|C|+|C_0|+n|C_i|+|tt| \tag{1}$$

In cloud computing, assume the length of time is 2 Byte, the bilinear mapping is Tate pairing, and the rank of p is 20 Byte prime number. Then, the total length of ciphertext is $2+(n+2)|p|$. When access structure value $n=3$, energy consumption computation is shown in Table 5 and 6.

Table 5. Comparison of energy consumption between the proposed scheme and FABSC scheme

Length of *P*	Energy consumption of the proposed scheme /mJ	Energy consumption of FABSC scheme/mJ
P = 20Byte	8.956	9.14
P = 42.5Byte	18.834	19.02
P = 60Byte	26.516	28.46

Table 6. Comparison of energy consumption between the proposed scheme and other schemes

Length of *P* and scheme	Energy consumption/mJ
P = 20Byte	8.956
P = 42.5Byte	18.834
P = 60Byte	26.515
Certificate-based	146.98
Merkle hash tree	144.57
ID-based	111.03

The computation of Tate pairing requires about 752ms, and the power is 62.04W. In this scheme, decryption needs n+1 times Tate pairing. Assume $n=3$, the power is 4×62.04W=248.16W. Table 7 shows comparison of control power. From Table 7, the energy consumption of the proposed scheme is higher than others. However, the communication energy consumption is far lower than others. Because the communication energy consumption is much higher than the computation energy consumption, the overall energy consumption is less than other schemes. It shows that the flexibility of access control can be improved effectively by using this method.

Table 7. Comparison of control power between the proposed scheme and other schemes

scheme	power/W
Certificate-based	39.96
Merkle hash tree	18.49
ID-based	124.07
FABSC	310.3
The proposed method	248.17

4. Conclusions

A ciphertext access control method CACDP is proposed for dynamic policy update in cloud computing. Based on the binary Trie tree, the key derivation mechanism is introduced, and the Keyder-Trie key management tree is constructed to further reduce the complexity of the DO maintenance key and improve the key security. The CACDP method uses the ELGamal-based proxy re-encryption algorithm to transfer the key re-encryption task caused by the access control strategy update to the CSP, thus reducing the cost of key update. Meanwhile, a double layer encryption strategy is designed to minimize the return frequency of the data in the policy update. In this paper, the correctness and security of the scheme are proved theoretically through security analysis. Finally, experiments show that our proposed scheme can obviously reduce the performance cost of policy update and improve the flexibility of access control.

Acknowledgements

This research is supported by Shenzhen scientific and technological innovation project (2017 No.272).

References

1. F. I. K. Mousa, N. Al Maadeed, K. Busawon, A. Bouridane, and R. Binns, "Secure MIMO Visible Light Communication System based on User's Location and Encryption," *Journal of Lightwave Technology*, Vol. 35, pp. 5324-5334, 2017
2. F. Guo and T. Cheng, "Systems and Methods for Detecting Suspicious Attempts to Access Data based on Organizational Relationships," Google Patents, 2015

3. M. Kumar, S. Verma, and K. Lata, "Secure Data Aggregation in Wireless Sensor Networks using Homomorphic Encryption," *International Journal of Electronics*, Vol. 102, pp. 690-702, 2015
4. A. Argyris, E. Pikasis, and D. Syvridis, "Gb/s One-Time-Pad Data Encryption with Synchronized Chaos-based True Random Bit Generators," *Journal of Lightwave Technology*, Vol. 34, pp. 5325-5331, 2016
5. Z. Pan, S. Liu, and W. Fu, "A Review of Visual Moving Target Tracking," *Multimedia Tools & Applications*, Vol. 76, pp. 16989-17018, 2017
6. H. Chen, X. Du, and Z. Liu, "Optical Hyperspectral Data Encryption in Spectrum Domain by using 3D Arnold and Gyrator Transforms," *Spectroscopy Letters*, Vol. 49, pp. 103-107, 2016
7. B. Cui, Z. Liu, and L. Wang, "Key-Aggregate Searchable Encryption (KASE) for Group Data Sharing via Cloud Storage," *IEEE Transactions on Computers*, Vol. 65, pp. 2374-2385, 2016
8. S. Liu, W. Fu, W. Zhao, J. Zhou, and Q. Li, "A Novel Fusion Method by Static and Moving Facial Capture," *Mathematical Problems in Engineering*, Vol. 2013, pp. 503924, 2013
9. M. Du and G. Hao, "DR-PRO: Cloud-Storage Privilege Revoking Optimization Mechanism based on Dynamic Re-Encryption," *Journal of Computer Applications*, Vol. 7, pp. 20, 2015
10. G. Wang, Q. Liu, and J. Wu, "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, 2010
11. G. Liang, "Big Data Encryption to Protect Privacy Data Environment Improvement and Simulation of the Algorithm," *Computer Simulation*, Vol. 33, pp. 338-341, 2016
12. Y. Tian, Y. Peng, X. Peng, and H. Li, "An Attribute-based Encryption Scheme with Revocation for Fine-Grained Access Control in Wireless Body Area Networks," *International Journal of Distributed Sensor Networks*, Vol. 10, pp. 259798, 2014
13. S. Liu, W. Fu, H. Deng, C. Lan, and J. Zhou, "Distributional Fractal Creating Algorithm in Parallel Environment," *International Journal of Distributed Sensor Networks*, Vol. 9, pp. 281707, 2013