Enhancing Cloud Computing Security and Privacy by Sharing Data Based on Attributes

Nyalapally Navneetha[1]

Sayyad Rasheed Uddin[2]

[1]MTech Student, Department of CSE, Malla Reddy College of Engineering, Dulapally Road Maisammaguda, Hyderabad, Telangana 500100, kokkulanavaneetha@gmail.com

[2]Assistant professor, Malla Reddy College of Engineering, Dulapally Road, Maisammaguda, Hyderabad, Telangana 500100, rasheeduddin.mrce@gmail.com

Abstract: Cloud computing makes it easy and affordable to share data. As a result, the data's contents are also protected since the data is stored on cloud servers. As a means of better securing shared data, many methods are used in order to safeguard its value and sensitivity. Cipher text-policy attribute-based encryption (CP-ABE) may be used in these ways to make it easier and more secure. Traditional CP-ABE focuses only on data confidentiality, ignoring the need of protecting the user's personal privacy. Confidentiality is ensured by CP-ABE with a disguised access policy, which also protects the user's privacy. On the other hand, most current approaches are inefficient due to their high communication and computing costs. Furthermore, authority verification and the issue of privacy leakage during the period of authority verification are not taken into account in the majority of these efforts. CP-ABE with efficient authority verification is proposed in this study to address the issues raised above. Furthermore, the size of the secret keys remains constant. When considering n-BDHE and decisional linear assumptions, however, a suggested approach provides selective security. The given scheme's merits have been validated by computer simulations.

Key Terms: Attribute-based encryption (ABE), authority verification, hidden access policy, privacy preserving

I. Introduction

Utilizing IT resources in the corporate realm is now accessible thanks to cloud computing approaches. Various scalable services are available on demand, such as online databases, programme interfaces, storage and computing resources, and so on, via the cloud. Fig. 1 shows how users may access services through phones, laptops, and desktops. It is possible to store and manage data remotely using cloud storage. Because it can simultaneously deliver a wide range of services, it is also useful for data analysis and computation. Data storage on the cloud offers various benefits, including lower communication and maintenance costs, resource savings, and the ability to access data from anywhere. Data confidentiality and privacy concerns may deter consumers from using cloud storage, despite the numerous advantages that come with it. Because the cloud service provider may get and expose users' personal privacy, and even access and share data unlawfully, if data is uploaded to cloud, the cloud service provider (CS) may be untrustworthy [1].

To make sure the confidentiality of the data in cloud, people are inclined to encrypt them before they are uploaded to cloud. But the general encryption algorithms make the data process become difficult. ABE is a good candidate to overcome this limitation. ABE was first proposed in 2005 by Sahai and Waters [2], which guaranteed the data confidentiality and provided the fine-grained access control policy to the customers. It has been widely accepted as an effective method encrypting the outsourced data in cloud computing. ABE improves the efficiency when the data owner (DO) intends to share data contents with multiusers. It permits DO to specify an access policy to the encrypted files, which can make the users who match it, access uploaded data. The users who do not satisfy the access structure cannot get any information about the data contents. For instance, we consider the data access control for a company. If the CEO intends to submit a classified file, through the cloud, to the managers in sales department, planning department, and research and development (R&D) department. Then he/she can use an ABE scheme. First, he/she encrypts the file and specifies an access structure as ω = manager ∧ (sales department ∨planning department ∨ R&D). Next, he/she uploads the encrypted file and the access structure into the CS. Only the managers in the three mentioned departments can access the classified file, and the managers in other departments or the general staff in the three mentioned departments cannot learn anything about the file even if they collude.

Most of ABE proposals perform very well in secure data sharing. However, the personal privacy of the DO and the users is ignored in these constructions. For convenience of recovering data, the access policy is always sent with ciphertexts. In some scenarios, the access structure may carry sensitive information of users. For instance, a patient wants to share his/her personal health record (PHR) with some doctors and family members, but he/she may not want others to know that he/she is sick. If the patient employs a normal ABE scheme to encrypt the PHR, although the malicious user cannot get the contents of the PHR, he/she may get some information about the users as shown in Fig. 2. The access policy contains “cardiopathy” and “DC hospital” and the malicious third party may guess that the

DO is suffering from a heart attack and is treating in the DC hospital. Hence a natural problem is how to keep the shared data secure, while the privacy of them is also protected.

II. Literature survey

In cloud computing, there have been many of the schemes, offered for encryption. Such as simple encryption technique that is classically studied. ABE scheme has been developed and modified further into Key Policy Attribute-based encryption (KP-ABE), CP-ABE [10].

Sahai et al. [11] in 2005 introduced Fuzzy identity-based encryption (IBE) which is seminal work of attribute-based encryption. After that in 2006, they [4] first proposed the attribute-based encryption. In ABE scheme both the secret user key and the ciphertext are associated with a set of attributes. A user can able to decrypt the ciphertext if and only if at least a threshold number of attributes matches associated with the ciphertext and user secret key. Different from traditional public key cryptography such as Identity-Based Encryption, ABE is implemented for one-to-many encryption in which ciphertext is not necessarily encrypted to one particular user, it may be for more than one number of users. In Sahai and Waters ABE scheme, the threshold semantics are not very expressive to be used for designing more general access control system. ABE in which policies are specified and imposed in the encryption algorithm itself. The existing ABE schemes are of two types KP-ABE scheme and CP-ABE scheme. V.

Goyal et al. [12] in 2006 introduced a KP-ABE scheme. It enables more general access control. It is the modified approach of a general model of ABE that has discussed earlier. Exploring KP-ABE scheme, attributes are associated with ciphertext and access policies related to secret keys of users. For decrypt the ciphertext, an access policy associated with user's secret key that is to be satisfied by the attributes associated with the ciphertext. KP-ABE scheme follows a public key encryption technique that is intended for one-tomany communications. For example, let us assume that the universe of attributes is defined as {A, B, C, D}. The ciphertext is computed using the set of attributes {A, B}. An access policy (A ⋀ C) ⋁ D is implanted into user's secret key. In this above example, the user would not be able to decrypt the ciphertext but would able to decrypt a ciphertext concerning attributes {A, C, D}. Sahai et al. [13] introduced the concept of another improved form of ABE called CP-ABE. In CP-ABE scheme, attribute policies are correlated with the ciphertext and attributes are associated with user’s secret keys and only those keys that the associated attributes satisfy the policy associated with the data can decrypt the ciphertext. CP-ABE works in the opposite manner of KP-ABE. While encrypting a plain text, the encrypter specifies the threshold access policy for his interesting attributes. After that plaintext is encrypted based on specified access policy, only those users whose attributes stored in secret key satisfy the access policy can decrypt the ciphertext. For instance, let us assume that the universe of attributes is defined as {A, B, C, D}, and user1 receive a secret key to attributes {A, B} and user2 to attribute {D}. If a ciphertext is encrypted concerning the policy $(A \land C) \lor D$, then user2 will be able to decrypt, while user1 will not be able to decrypt. With CP-ABE technique, encrypted data can be kept confidential even if the storage server is un-trusted and more secure against collusion attacks. CP-ABE scheme is more natural to apply instead of KP-ABE to enforce access control of encrypted data. Almost all existing CP-ABE system requires full trusted authority.

M. Chase et al. [14] in 2009 introduced a distributed KP-ABE scheme to solve the key escrow problem in a multi-authority system. In this scheme, all authorities are participating in the key generation protocol in a distributed way considering they have not colluded with each other. Because there is no centralized trusted authority with master secret information, all attribute authorities should communicate with others in the system to create a user's secret key. A primary concern of this approach is the performance degradation. It results in O(N 2) communication overhead on both the system setup phase and any rekeying phase. In addition to the attribute keys, each user requires storing O(N 2) additional auxiliary key components, where N is the number of parties in the system.

J. Hur [15] in 2013 provided an improved security data sharing scheme based on the classic CP-ABE. The key escrow issue is addressed by using an escrow-free key issuing mechanism where the key generation center and the data storage center work together to generate secret key for user. The protocol requires interactive computation between the both parties. So, the computational cost in generating user’s secret key increases. The performance and security analyses indicate that the proposed scheme is efficient to securely manage the data distributed in the data sharing system.

X. Xie et al. [16] in 2013 presented a novel access control scheme in cloud computing with efficient attribute and user revocation. The computational overhead is significantly eliminated from O(2N) to O(N) in user key generation by improving CP-ABE scheme, where N is the number of attributes. The size of ciphertext is approximately reduced to half of original size of plaintext. However, the security proof of the scheme is not fully given. Most of the existing CPABE schemes require a full trusted authority with its master secret key as input to generate and issue the secret keys of users. Thus, the key escrow issue is inherent, such that the authority has the "power" to decrypt all the ciphertext of system users [17].

Fan et al. [18] in 2014 proposed an arbitrary-state ABE to solve the issue of the dynamic membership management. This paper provides high flexibility of the constraints on attributes and makes users be able to dynamically join, leave, and update their attributes. A user is allowed to enroll and leave from an ABE system, and she/he can also change her/his attributes and the values corresponding to the attributes. It is unnecessary for anyone else to update her/his private key when enrollment, leaving, or attribute updating occurs.

III. Proposed methodology

Privacy Preservation Access Control Mechanism in Cloud

Group Key Management is a technique for safely allocating a message to a user group with confidentiality as the main key. Here users in a group has a symmetric key K, called the group key, and when they have to share a message, it must be encrypted with K and sent to the members of a group. As K is known to all members in the group, they can decrypt and get the message. When new user enters or exits a group, a new group key is created and distributed. No new member can get access to earlier transmitted messages (backward secrecy), and no user who has left the group can learn anything from future communications in the group (forward secrecy). This is the reason for rekeying when the user dynamics changes.

This approach is not desirable if there are frequent leaves and joins and in broadcast group key management scheme for distributing keys to users, each user gets one or more secrets that is combined with some variable to get the group key. With change in group dynamics changes there will be change in public information only. The encrypted data is uploaded to the cloud and each user is given the keys only for the set(s) of data items that it can access according to the policies 1. This approach protects data confidentiality and enforces fine grained access control policies. Matter of concern is due to key management, as every user must be given the correct keys with respect to access control policies the user satisfies. The solution is to provide a hybrid solution by encrypting keys with attribute-based encryption and/or proxy reencryption which also has lot of weakness.

Attribute based group key management uses the following algorithms:

1. Initial_Setup algorithm that uses the security parameter that initializes all parameters including set of secrets U, secret space US and key space Ks
2. Secret_Generator(user,att) algorithm picks random string s from U at random from US, adds s to U and outputs s. For every user, a secret key is assigned which is used for generating group key.
3. Key_Generator(U,policy) algorithm picks group key g from key space Ks and generates tuple GT that is calculated from U
4. Key_derivative(s,GT) algorithm uses U and GT to get group key 5.
5. Update(U) algorithm executes Key_Generator algorithm when there is entry or exit in group.

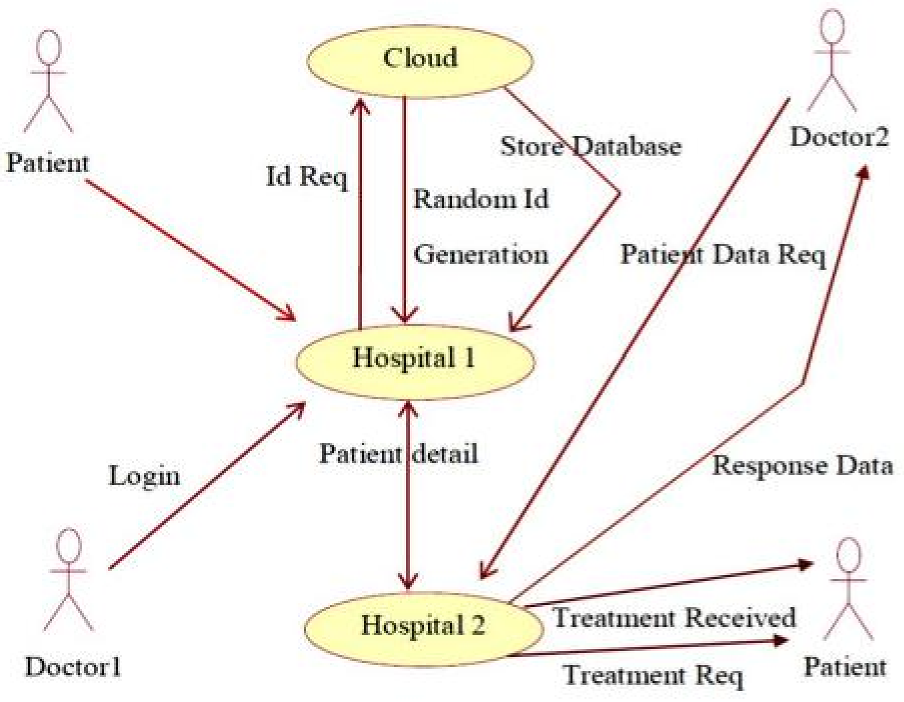

Fig 1: Information flow in hospital cloud

In a typical hospital scenario in figure 2 that plans to utilize cloud for maintaining the sensitive health record of patients, the confidentiality must be preserved in cloud. The stakeholders in hospital are doctor, pharmacist, receptionist, cashier, nurse, data entry operators etc. Each stakeholder need different level of access of detail and so the health record maintenance need fine-grained access control. Attributes like role, position and location are used as identity attributes which are grouped properly so that there is no privacy violation. The double encryption method for achieving fine grained access control has six phases for the whole process which are:

1. Token provider issues tokens to user based on attribute identity. Access control policy is decomposed between the data owner and cloud.
2. The user registers their identity tokens with owner and cloud.
3. Secrets are shared between user, data owner and cloud.
4. Data owner selectively encrypts and uploads keeping one set of tokens to itself and another set to cloud preventing cloud alone to decrypt the data.
5. Re-encrypt and enforce policy
6. Encrypted data is downloaded from cloud and by using derived keys, decryption is done twice.

The flow of events in the system as shown in fig.1 is explained below Registration in User Enrollment: All users in the system has to be enrolled in Registration and they get an Id Number. New Patient Registration: Each new patient has to be registered in hospitals with an id. Create New Registration: If the user is new employee in hospital, then create a new registration with new Id. Create New Id Generate and Fill Appointment Form: The user creates a new Id and Password and fill the Various Information in the Appointment form Details.

System Overview

In this proposed System, 1. PHR owner initially authorizes a Central authority (CA). 2. Central Authority is responsible for generating Master key (MK) and Secret Key (SK). 3. CA issues MK and SK to user. 4. PHR owner will Encrypt the file using ABE. 5. Encrypted PHR will be outsourced to cloud server. 6. Meanwhile users provides attribute value to owner. 7. User gets the write key from PHR owner. 8. The user can read or write file by providing their respective access policies and secret keys. 9. The encrypted file search is done using the Bloom filter.

Encryption of Phr Using Abe

This module deals with the encryption PHR's by data owner. The MAABE scheme is executed as follows and comprises of six algorithms, Actaul Setup: The setup algorithm takes no input other than the implied security parameter. This creates the public parameters PK and a master/secret key MK produced by each CAs.The N-th CA terms a disjoint set of role attributes Ur, which are equally common for public users. As these attributes are characterized on the basis of their profession. Key generation (MK, SK): The key generation algorithm uses the master key MK and a set of attributes Ur that describe the key, and outputs a secret key SK for user U. SK shall contain at least one attribute from every type of attributes governed by CA.

Encryption (PK, M, Ur):

1. The encryption algorithm takings the public parameters PK as input, a message, M and an access structure A over a set of attributes Ur.

2. It will encrypt M, and produce a cipher text CT such that only a user who keeps the set of attributes sustaining the access structure will be able to decrypt CT. Create user (PK, MK, U): The Create User algorithm takes as input the public key PK, the master key MK, and a user name U. It outputs a public user key for user i.e. PK, This key will be used by CA to issue secret keys for user U .

The secret user key SK, can be used for the decryption of cipher texts. Create authority (PK, a): The Create Authority algorithm is executed by the admin with identifier a once during initialization. It outputs a secret authority key SKa for the attribute authority.

Request Attribute SK (PK, a, SK, U): The Request Attribute SK algorithm is executed by the attribute authority whenever it gets a request for a secret attribute key. The algorithm verifies whether the user U with public key PK is eligible for set of role attributes. If the condition satisfies Request Attribute outputs a secret key SK for the user U is the algorithm returns Null value.

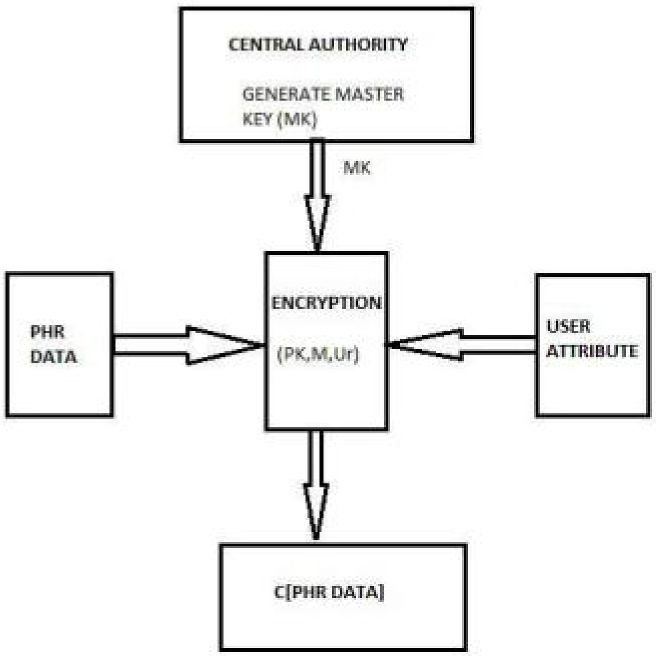

Fig 2: Encryption of PHR's data

This module describes how the PHR data are getting stored in the cloud. At first, we have to setup a private cloud storage using Eucalyptus. After the cloud installation, the encrypted PHR data are sent to the cloud storage using APIs like S3curl, S3cmd and S3fs.

The Bloom Filter is actually a data structure which shall store the fundamentals of a set in a space in a well-organized manner considering that a small error is permissible while testing for elements in the Bloom Filter. The data structure is basically used for membership queries. Its main properties are: The quantity of space required to store the bloom filter is very small as matched to the data. The actual time required to validate whether an element is existing or not is independent of the number of elements which are actually present in the set.

The Bloom filter uses an array of m-bits which are all initially set to 0 and k hash functions each returning a value between 1 and m, to set and check these bits.

1. For storing the given element e in bloom filter b, it is applied to each hash function (h1, h2,., hk) and based on the return value v of each function (v1, v2, vk) the bit with offset v is set to 1. Since there are k hash functions, up to k bits are set to 1 since some hash functions may return the same value.

2. To check for an element whether it's present, again the bloom filter is applied on each hash function and checked whether any of the offset are set to 0. If yes, this implies that the element is not present. If all bits are set to 1 it implies that element may be present". Figure 3.3 shows Bloom filter set and checking operation.

Decryption of PHR Decryption (PK, CT, SK): The decryption algorithm takes as input PK, a cipher text CT, which was obtained for set of attributes Ur, and a private key SK for Ur. If Ur gratifies the access structure A, then the algorithm will decrypt the cipher text and return a message M" shown Figure 3.

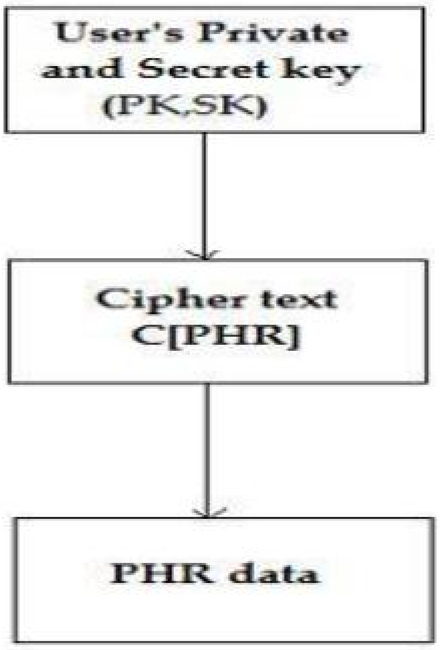

Fig 3: Decryption of PHR data

Overall System Flow

In this section the overall system flow is depicted. Figure 3.7 shows the overall system flow.

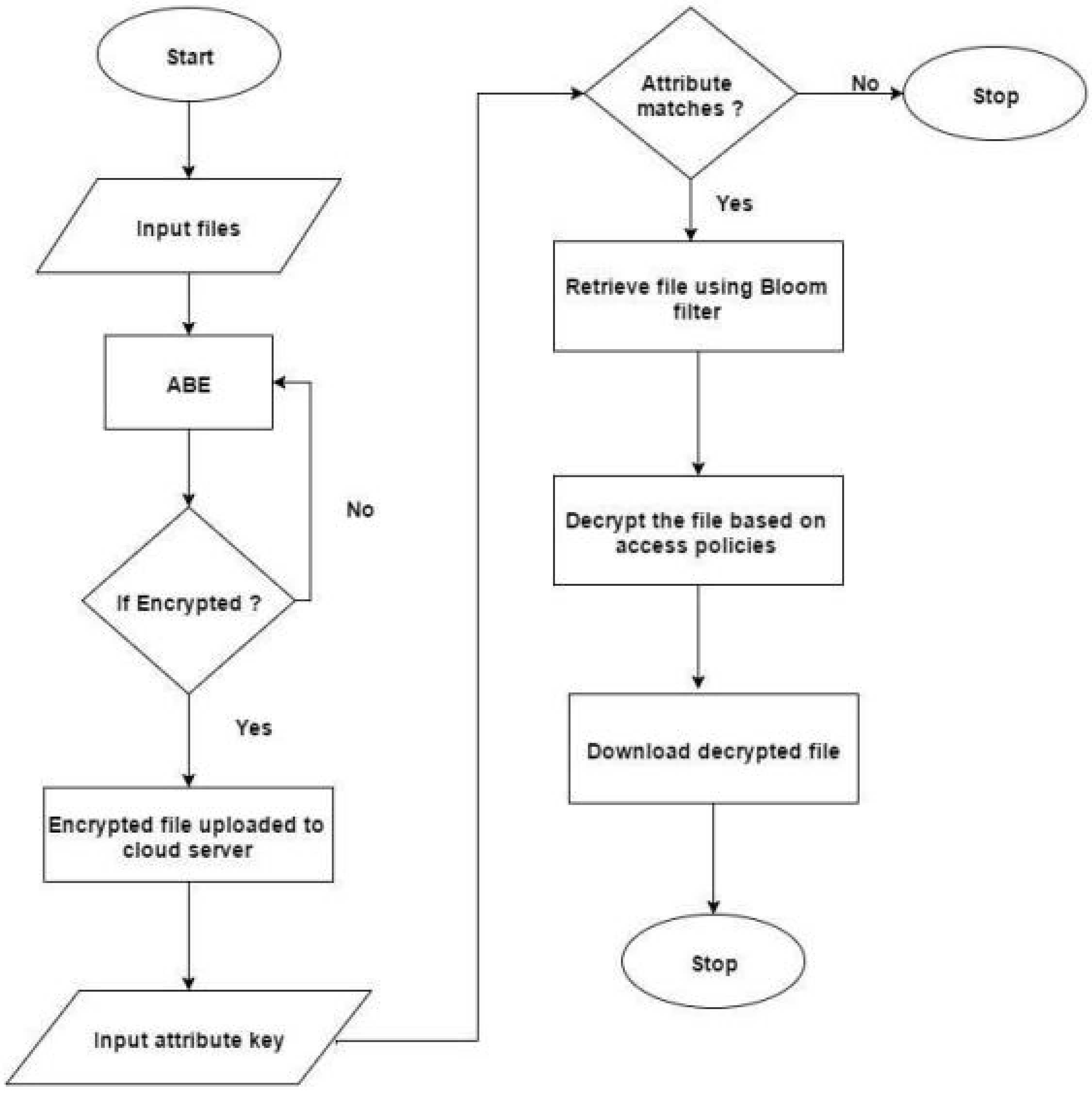
Start
Input files
ABE
If Encrypted ?
No
Yes
Encrypted file uploaded to cloud server
Input attribute key
Attribute matches ?
No
Stop
Yes
Retrieve file using Bloom filter
Decrypt the file based on access policies
Download decrypted file
Stop

Fig 4: Overall System Flow

IV. Conclusion

We proposed a privacy preserving CP-ABE scheme in the standard model. The presented scheme has many advantages over the existing schemes, such as constant size private keys and short cipher texts. And in decryption, it only needs four pairing computations. The proposed scheme achieves selective security and anonymity in a prime order group. In the standard model, we show the security of the proposed scheme is reduced to the decisional *n*-BDHE and the DL assumptions. Additionally, the proposed scheme supports authority verification with no privacy leakage.

However, the introduced scheme only supports "AND" policy and relies on a weak security model. How to construct a strong secure HP-CP-ABE scheme with more flexible access policy is left for the future works.

References

[1] P. P.Kumar, P. S.Kumar, and P. J. A. Alphonse, "Attribute based encryption in cloud computing:Asurvey, gap analysis, and future directions," J. Netw. Comput. Appl., vol. 108, pp. 37–52, 2018.

[2] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in Proc. 24th Annu. Int. Conf. Theory Applications Cryptographic Techn., May 2005, vol. LNCS 3494, 2015, pp. 457–473.

[3] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertextpolicy attribute-based encryption scheme with constant ciphertext length," in Proc. 5th Int. Conf. Inf. Security Practice Experience,Apr. 2009, pp. 13–23.

[4] J. Han, W. Susilo, Y. Mu, and J. Yan, "Privacy-preserving decentralized key-policy attribute-based Encryption," IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 11, pp. 2150–2162, Nov. 2012.

[5] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," IEEE Trans. Inf. Forensics Secur., vol. 11, no. 6, pp. 1256–1277, Jun. 2016.

[6] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in Proc. 30th Annu. Int. Conf. Theory Appl. Cryptographic Techn.: Advances Cryptology, May 2011, pp. 568–588.

[7] B.Waters, "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions," in Proc. 29th Annu. Int. Cryptology Conf. Advances Cryptology, Aug. 2009, pp. 619–636.

[8] T. Nishide, K. Yoneyama, and K. Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in Proc. Appl. Cryptogr. Netw. Security, Jun. 2008, vol. LNCS 5037, pp. 111–129.

[9] J. Lai, X. Zhou, R. H. Deng, and Y. Li, "Fully secure cipertext-policy hiding CP-ABE," in Proc. 6th ACM Symp. Inf. Comput. Commun. Secur.,2011, pp. 24–39.

[10] J. Lai, X. Zhou, R. H. Deng,Y. Li, and K. Chen, "Expressive CP-ABE with partially hidden access structures," in Proc. 7th ACM Symp. Inf. Comput. Commun. Secur., May 2012, pp. 18–19.

[11] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization,” in Proc. 14th Int. Conf. Practice Theory Public Key Cryptography Conf. Public Key Cryptography, Mar. 2011, pp 53–70.

[12] Y. S. Rao and R. Dutta, “Recipient anonymous ciphertext-policy attribute-based encryption,” in Proc. 9th Int. Conf. Inf. Sys. Secur., Dec. 2013, pp. 329–344.

[13] L. Zhang, Q. Wu, Y. Mu, and J. Zhang, “Privacy-preserving and secure sharing of PHR in the cloud,” J. Med. Syst., vol. 40, pp. 1–13, 2016.

[14] M. Abdalla, D. Catalano, and D. Fiore,“Verifiable random functions: Relations to identity-based key encapsulation and new constructions,” J. Cryptol., vol. 27, pp. 544–593, 2014.

[15] C. Huang, K. Yan, S.Wei, G. Zhang, and D. H. Lee, “Efficient anonymous attribute-based encryption with access policy hidden for cloud computing,” in Proc. IEEE Int. Conf. Progress Inform. Comput., Dec. 2017, pp. 266–270.

[16] Y. Zhang, X. Chen, J. Li, D.Wong, and H. Li “Anonymous attribute-based encryption supporting efficient decryption test,” in Proc. 8th ACM Symp. Inf. Comput. Commun. Secur., May 2013, pp. 511–516.

[17] J. Li, H. Wang, Y. Zhang, and J. Shen, “Ciphertext-policy attribute-based encryption with hidden access policy and testing,” KSII Trans. Internet Inf. Syst., vol. 10, no. 7, pp. 3339–3352, Jul. 2016.

[18] H. Cui, R. H. Deng, G. Wu, and J. Lai, “An efficient and expressive Ciphertext-policy attribute-based encryption scheme with partially hidden access structures,” in Proc. 10th Int. Conf. Prov. Secur., Nov. 2016, pp. 19–38.

[19] F. Khan, H. Li, L. Zhang, and J. Shen, “An expressive hidden access policy CP-ABE,” in Proc. IEEE 2nd Int. Conf. Data Sci. Cyberspace, Jun. 2017, pp. 26–29.

[20] Y. Zhang, Z. Dong, and R. H. Deng, “Security and privacy in smart health: Efficient policy-hiding attribute-based access control,” IEEE Int. Things J., vol. 5, no. 3, pp. 2130–2145, Jun. 2018.

[21] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption,” in Proc. 29th Annu. Int. Conf. Theory Appl. Cryptographic Techn., 2010, pp. 62–91.