Attribute-based Data Sharing Scheme in Cloud computing using Weight

Akshay Choudhari
PG Student
MEIT Department
PICT, Pune

Emmanual M., PhD
Professor
MEIT Department
PICT, Pune

ABSTRACT
To maintain data integrity on the cloud, Attribute-based Encryption (ABE) with Key Policy Attribute-based Encryption (KP-ABE) and Ciphertext-Policy Attribute-based Encryption (CP-ABE) can be used with access control implementation for cloud computing. CP-ABE is a promising cryptographic primitive for secure data sharing in cloud computing. A data owner is the only charge of to define the access policy associated with his data which to be shared. In CP-ABE, each user's secret keys are associated with a set of attributes and data are encrypted with access policy on attributes. A user can decrypt a ciphertext if and only if his attributes satisfy the ciphertext access policy. In CP-ABE, the secret keys of users have to be issued by a trusted key authority that leads to key escrow problem. Besides, most of the existing CP-ABE schemes cannot support attribute with an arbitrary state. In this paper, weighted-attribute data sharing scheme is proposed to solve the key escrow problem and also improve the expressiveness of attribute, so that the resulting scheme is friendlier to cloud computing applications. An improved two-party key issuing protocol guarantees that neither key authority nor cloud service provider can compromise the whole secret key of a user individually. The concept of weighted-attribute not only enhance the expression of an attribute binary to arbitrary but also reduce the complexity of access policy, so that storage cost of ciphertext and time cost in encryption can be reduced.

Keywords
Cloud Computing, Attribute-based Encryption, Secure Data Sharing, Weighted-Attribute

1. INTRODUCTION
Cloud Computing has become a booming research area due to its long-list advantages such as high scalability, convenience, cost saving, and disaster recovery. Cloud system famous for data sharing capability and this can provides plentiful benefits to the user. There is currently a push for IT organizations to increase their data sharing efforts. According to a survey by InformationWeek, almost all IT organizations shared their data. The 74 % organization share their data with customers and 64 % organization sharing with suppliers. Data sharing is a top priority of every fourth of surveyed organizations [1], [2].

The Cloud, however, is liable to many privacy and security attacks. The biggest difficulty hampering the progress and the wide adoption of the Cloud is the privacy and security concerns associated with it. According to a survey carried out by IDC Enterprise Panel in August 2008, Cloud users look upon security as the top challenge with 75 % of surveyed users bothered about their critical business and IT systems being susceptible to attack. Many security and privacy attacks occur from within the Cloud service provider themselves as they usually have straightforward access to stored data and steal the data to sell to third parties to gain commercial benefits [3], [4].

Therefore, data security and privacy are the most important concern in cloud computing. Cryptography in the cloud provides encryption techniques to secure data that will be used or stored in the cloud. It allows users too conveniently and securely accesses shared cloud services, as any data that is stored in cloud storage is protected with encryption. Cryptography techniques in the cloud computing protect sensitive data without delaying information exchange. In security enforcement of information system, an access control is one of most commonly used approach. Access control is a policy that permits, rejects or confines access to the resources in a computing environment. It also monitors and records all attempts made to access a system. It is a mechanism which is very much important for protection in computer security. Accordingly, how to efficiently and securely share user data is one of the toughest challenges in the scenario of cloud computing. This proposed work revised on Attribute-Based Encryption methods which have been developed so far for achieving secure data sharing in cloud computing [5], [6].

CP-ABE has become to be a vital encryption technology to tackle the challenge of secure data sharing. In a CP-ABE, user's secret key is defined by an attribute set, and ciphertext is associated with an access structure. Data Owner (DO) is allowed to determine access structure over the universe of attributes. A user can able to decrypt a given encrypted text only if his/her attribute collection set matches the access policy over the ciphertext. Employing a CP-ABE system directly into a cloud application that may yield some open problems. Firstly, all secret keys of users need to be issued by an entirely trusted key authority (KA). This causes a security hazard that is known as key escrow problem. By knowing the secret key of a user, the KA can able to decrypt all the user's ciphertexts, which is in total against to the will of the user. Secondly, the expressiveness of attribute set is another concern. The existing CP-ABE schemes can only define binary state over attribute, for example, 1for satisfying and 0 for not-satisfying, but not dealing with the arbitrary-state attribute [7], [8], [9].

In this proposed work, the weighted attribute is introduced to not only improve attribute expression from binary to arbitrary nature, but also to simplify access policy. Thus, the storage cost and encryption cost of ciphertext can be relieved. The proposed work successfully resolves two types of problems: key escrow and arbitrary-sate attribute expression.

The proposed work includes following:

• An improved key issuing protocol to resolve the key escrow

problem of CP-ABE in cloud computing. The protocol can prevent KA and CSP from knowing each other's master secret key so that none of them can create the whole secret keys of users individually. Thus, the fully trusted KA can be semi-trusted. Data confidentiality and privacy can be ensured.

• Weighted attribute to improve the expression of the attribute. The weighted attribute can not only express arbitrary-state attribute (instead of the traditional binary state) but also reduce the complexity of access policy. Thus the storage cost of ciphertext and computation complexity in encryption can be reduced. Besides, it can express larger attribute space than ever under the same condition.

• The proposed work is efficient both regarding computation complexity and storage cost.

2. 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-to-many 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 attribute {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 ∧ C) ∨ 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 CP-ABE 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 papers 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.

3. PROPOSED SYSTEM

3.1 Access Policy

Suppose there is a formal structure in college, in which teachers are grouped into teaching assistant, lecturer, associated professor and full professor. The weight of the

attribute for each kind of the teachers as 1, 2, 3, and 4. Hence, these attributes can be indicated as "Teacher: 1", "Teacher: 2", "Teacher: 3" and "Teacher: 4", respectively. In this instance, they can be denoted by one attribute which has just different weights. In particular, it can be arbitrary-state attributes, such as "Teacher: teaching assistant, lecturer, associate professor, full professor." Assume that an access structure is represented as: {("Lecturer" OR "Associate Professor" OR "Full Professor") AND "Male"}, and the existing CP-ABE systems are executed on the form of access policy. If proposed work is used, the access policy can be simplified as *T*' {"Teacher: 2" AND "Male"}, because the attribute "Teacher: 2" denotes the least level in the access structure and includes {"Teacher: 2", "Teacher: 3" "Teacher: 4"} by default. Therefore, the storage overhead of the related ciphertext and the computational cost used in encryption can be reduced. These two structures are shown in Fig. 1. Also, proposed work can be used to represent larger attribute space than ever under the same number of attributes. For example, if both the attribute space and weighted set include n elements, the proposed work can describe n^2 different possibilities. In contrast, the existing CP-ABE schemes only show 2n possibilities [19].

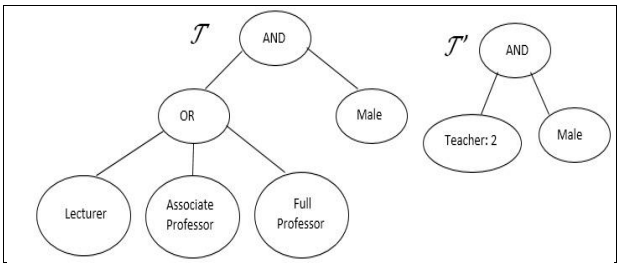

Fig 1: Access Policy Structure

3.2 System Model

As illustrated in Fig. 2 and Fig. 3, the system model and framework of proposed work in cloud computing are given, where the system consists of four types of entities: KA, CSP, DO and Users.

3.2.1 Key Authority (KA)

It is a semi-trusted key authority that generates public and secret parameters for CP-ABE. KA is responsible for issuing, revoking, and updating attribute keys for users. It also grants access rights to individual authorize users based on their attributes. It is assumed to be honest-but-curious. That is, it will fairly execute the assigned tasks in the system; however, it would like to learn information of encrypted contents as much as possible. Thus, it should be deterred from accessing the plaintext of the encrypted data even if it is honest.

3.2.2 Cloud service Provider (CSP)

It is an entity that provides a data sharing service. It is responsible of controlling the accesses from outside users to the storing data and providing corresponding contents services. The CSP is another semi-trusted key authority that generates personalized user key with the KA, and issues and revokes attribute group keys to valid users per each attribute, which are used to enforce a fine-grained user access control. Similar to the previous schemes, assume the CSP is also semi-trusted (that is, honest-but-curious) like the KA.

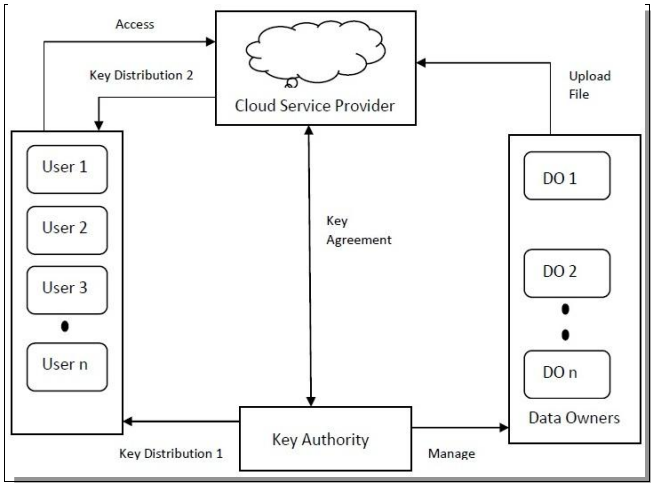

Fig 2: The Proposed Work model

3.2.3 Data Owner (DO)

It is an entity who owns data, and desired to upload it into the CSP for ease of sharing or for cost saving. The main work of data owner is to define (attribute-based) access policy, and imposing it on its own data by encrypting the data under the policy before distributing it.

3.2.4 Users

It is an entity who wants to access the data. If a user possesses a set of attributes matches the access policy of the encrypted data, and is not revoked in any of the valid attribute groups, then he will be able to decrypt the ciphertext and he is able to download the data.

Since both of the key distributing center, the KA and the CSP, are semi-trusted, they should be prevented from accessing plaintext of the data to be shared; for the moment, they should be still able to issue secret keys to users. The two parties involve in the arithmetic 2PC protocol with secret master keys of their own, and issue independent key components to users during the key issuing phase. The 2PC protocol deters them from knowing each other's master secrets so that none of them can generate the whole set of secret keys of users independently. Therefore, it takes an assumption that the KA does not collude with the CSP since they are honest.

3.3 Procedure

The proposed work contains the following four Modules:

3.3.1 System Initialization

This phase includes both entities: KA and CSP. Firstly, the system initialization setup invoke by Key Authority. For KA.Setup algorithm, the probabilistic operation inputs a security parameter κ. It generates a public parameter PP1 and a master secret key MSK1. After KA, system initialization setup invokes by the CSP. For CSP.Setup algorithm, it requires a security parameter κ as an input and generates PP2 and MSK2. The public parameter of the system is denoted as PP= {PP1, PP2} and master secret key of the system is denoted as MSK= {MSK1, MSK2}, where MSK1 and MSK2 are stored by KA and CSP, respectively.

3.3.2 Data Encryption

To improve the efficiency of encryption, DO first encrypts file M with content key ck by using simple symmetric encryption algorithm, where file ciphertext is denoted as Eck(M). Then, the content key ck is encrypted by the following procedure. The data owner invokes DO.Encrypt (PP,ck,A) algorithm.

This algorithm inputs PP, ck, and an access policy and it is executed by DO. It encrypts ck and outputs content key ciphertext CT which implicitly contains. Then, DO delivers Eck(M) and CT to CSP.

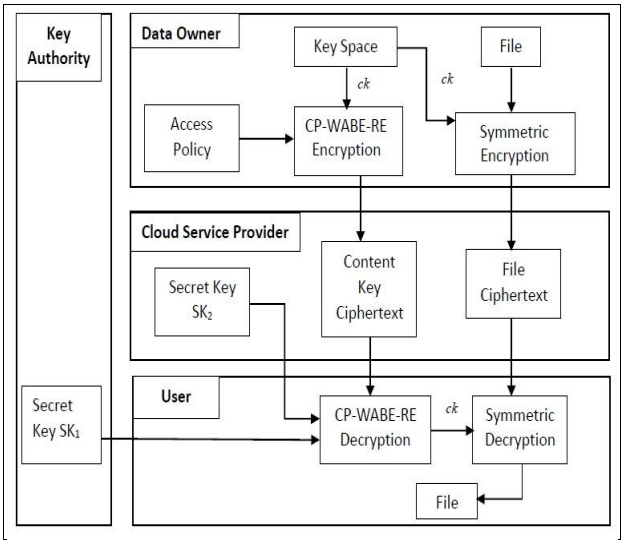

Fig 3: The Proposed framework

3.3.2 Data Encryption

To improve the efficiency of encryption, DO first encrypts file M with content key ck by using simple symmetric encryption algorithm, where file ciphertext is denoted as Eck(M). Then, the content key ck is encrypted by the following procedure. The data owner invokes DO.Encrypt (PP,ck,A) algorithm. This algorithm inputs PP, ck, and an access policy and it is executed by DO. It encrypts ck and outputs content key ciphertext CT which implicitly contains. Then, DO delivers Eck(M) and CT to CSP.

3.3.4 User Key Generation

This phase consists participation of KA and CSP. Firstly, in User key generation KA invokes KA.KeyGen(MSK1,S) algorithm. KA inputs MSK1 and a set of weighted attributes S. It creates secret key SK1 described by S. The proposed system is an improved two-party key issuing protocol to remove escrow. KA and CSP perform the improved protocol with master secret keys of their own. Thus, none of them can create the whole set of secret keys of users individually. Meanwhile, assume that KA does not collude with CSP since they are honest (otherwise, they can obtain the secret keys of each user by sharing their master secret keys).

After that CSP invokes CSP.KeyGen(MSK2) algorithm. This algorithm inputs MSK2 and the required information. It outputs secret key SK2 by executing the following key generation protocol.

KeyComKA↔CSP (MSK1,IDt,r,MSK2) → (SK2): It is an interactive algorithm between KA and CSP. KA inputs MSK1, a user identity IDt and a personalized secret r. CSP inputs MSK2 and IDt. At last, only CSP generates a personalized key component SK2 for the corresponding user. Then, the user constructs the whole secret key SK with the key components separately receiving from KA and CSP, i.e. SK= {SK1, SK2}.

3.3.4 Data Decryption

This phase contains two algorithms. User initial downloads file ciphertext Eck(M) and content key ciphertext CT from CSP. If he satisfies conditions, he can get content key ck by calling Users.Decrypt algorithm. Then, he uses ck to further decrypt file M by using Data.Decrypt operation. For the decryption user first, invokes Users.Decrypt (PP, SK, CT) algorithm. The algorithm inputs PP, SK described by S, and CT which includes access policy. Only when the weighted attribute set S matches the access policy, the content key ck is obtained. After obtaing the content key user invokes Data.Decrypt (Eck(M), ck) algorithm. The algorithm inputs file ciphertext Eck(M) and ck. It outputs file M, using symmetric decryption algorithm.

4. RESULTS AND ANALYSIS

The proposed work is simulated using CPABE toolkit and Java Pairing-Based Cryptography library (JPBC). The following analyses are conducted using Java on the system with Intel i3-4005U CPU at 1.70 GHz and 4.00 GB RAM running Windows. Besides, all the simulation results are the mean of 5 trials. The Kilobyte (KB) is units of storage cost and Seconds for time.

4.1 Analysis of User Key Generation

The storage overhead and computation cost of secret user key are compared as plotted in figure 4 and figure 5. The y-axis denotes storage overhead or time cost of user's secret key. The x-axis denotes the number of weighted attributes in user's secret key. The number of attributes used in this experiment is N = {10, 20, 30, 40, 50}.

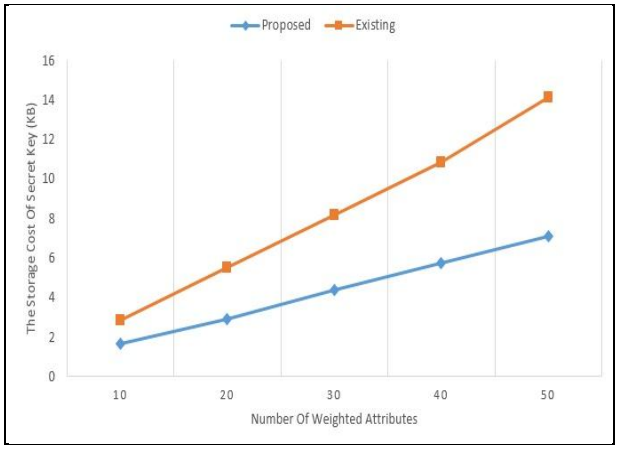

Fig 4: Comparison of Storage Cost of User Secret Key

The figure 4 shows the storage cost required for secret user key in both the schemes proposed work and existing scheme, arbitrary state attribute based encryption with dynamic membership under the different number of attributes. It can see the graph, storage cost for secret user key slowly rising and approximately follow a linear relationship with some attributes. Also, it is clear that the storage cost of secret user key using proposed work is smaller than the existing approach under the same number of attributes. Regarding percentage, the proposed work reduce average storage cost of secret user key by 87.58% as compared to existing approach.

The figure 5 shows the time cost required for user secret key generation in both the schemes, proposed work and existing scheme, arbitrary state attribute based encryption with dynamic membership under the different number of attributes. It can see from a graph, time cost for user secret key generation slowly rising and approximately follow a linear relationship with the some of the attributes. Also, it is clear that the time cost of user secret key generation using proposed scheme is lesser than the existing approach under the same

number of attributes. Regarding percentage, the proposed work reduce average time cost of user secret key generation by nearly half, i.e., 49.08% as compared to existing approach.

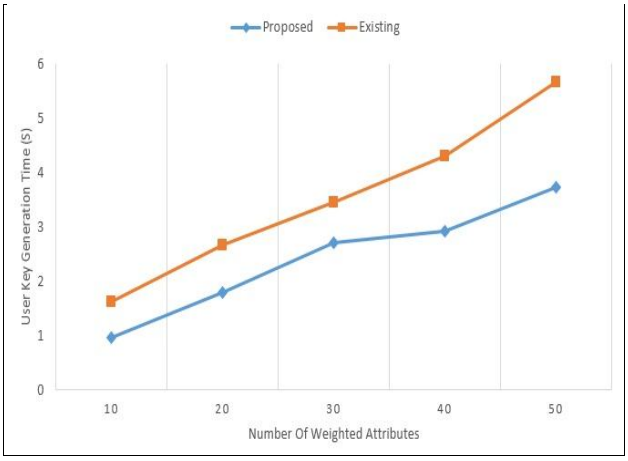

Fig 5: Comparison of Time Cost of User Secret Key Generation

4.2 Analysis of Data Encryption

The storage overhead and computation cost of encrypting data by a data owner are compared as plotted in figure 6 and figure 7. The y-axis denotes storage overhead or time cost of encrypting data by the data owner. The x-axis denotes the number of weighted attributes in access policy defined by the data owner. The number of attributes used in access policy is N = {10, 20, 30, 40, 50}.

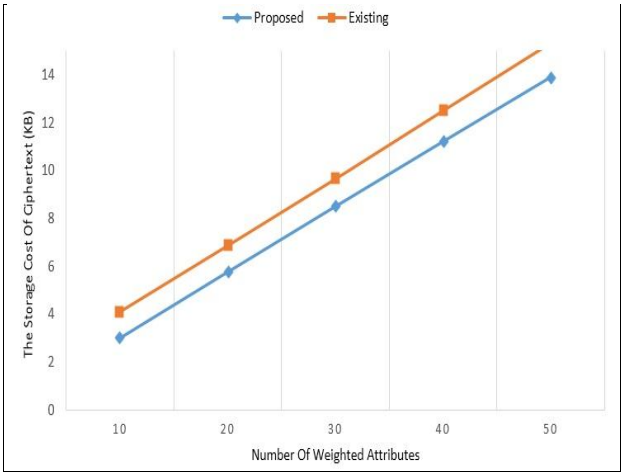

Fig 6: Comparison of Storage Cost of Ciphertext

The figure 6 shows the storage cost required for ciphertext in both the schemes, i.e., proposed work and existing scheme, i.e., arbitrary state attribute based encryption with dynamic membership under the different number of attributes. It can see from graph, storage cost for ciphertext gradually increasing and approximately follow a linear relationship with the number of attributes. Also, it is clear that the storage cost of ciphertext using proposed work is smaller than the existing approach under the same number of attributes. Regarding percentage, the proposed work reduce average storage cost of ciphertext by 17.87% as compared to existing approach.

The figure 7 shows the time cost required for data encryption in both the schemes, i.e., proposed work and existing scheme, i.e., arbitrary state attribute based encryption with dynamic membership under the different number of attributes.

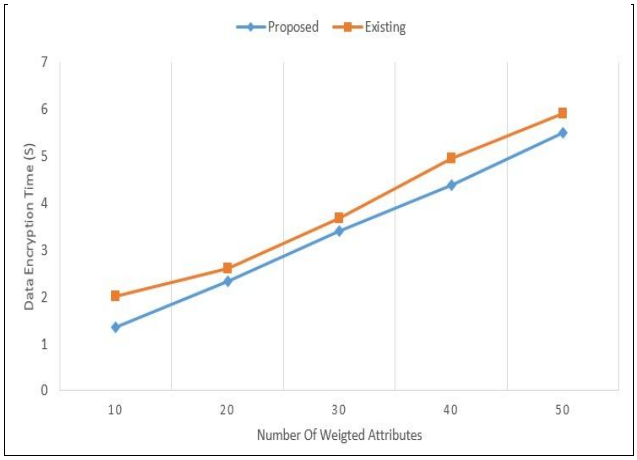

Fig 7: Comparison of Time Cost of Encryption

It can see from the graph, time cost for data encryption gradually increasing and approximately follow a linear proportion with the number of attributes. Also, it is clear that the time cost of data encryption using proposed work is lesser than the existing approach under the same number of attributes. Regarding percentage, the proposed work reduce average time cost of data encryption by 18.25% as compared to existing approach.

5. CONCLUSION

The proposed attribute-based data sharing scheme for a fine-grained data access control by exploiting the characteristic of the data sharing system. The secret user keys are generated through a secure two-party computation; this mechanism removes key escrow problem. It enhances privacy and data confidentiality in cloud system against the managers of KA and CSP as well as malicious system outsiders, making KA and CSP are semi-trusted. Also, the weighted attribute was proposed to improve the expression of the attribute, which can not only describe arbitrary-state attributes but also reduce the complexity of access structure, so that the storage cost of encrypted data and time cost in encryption can reduce. Therefore, the proposed work achieves secure and efficient data access control in the data sharing system. Also, it is dynamic and scalable to manage user data in the data sharing system securely. Finally, the graphs of experimental results describes comparison of proposed scheme against existing system. Therefore, proposed work take less time for encryption and save storage space of cloud service provider. The results demonstrate high efficiency and security of proposed work. The future scope of the proposed work can be searchable attribute based encryption and privacy preserving attribute based data sharing with proxy re-encryption. These are areas in which research is going on to leverage different technique to achieve data sharing.

6. ACKNOWLEDGMENTS

The authors are grateful to the Department of Information Technology of Pune Institute of Computer Technology (PICT) for providing their all support during this work. Authors are also grateful to the researcher for providing the research papers.

7. REFERENCES

[1] T. Grance and P. Mell, "The NIST Defination of Cloud Computing", NIST, U.S., 2011.

[2] InformationWeek, "Why IT Needs To Push Data Sharing Efforts InformationWeek," 2015. [Online]. Available:

http://www.informationweek.com/services/integration/why it needs to push data sharing effort/225700544.[Accessed: 27- Dec- 2015].

[3] Zhou M, Zhang R, Xie W, Qian W, Zhou A "Security and privacy in cloud computing: a survey." Sixth International conferences on Semantics knowledge and grid (SKG), pp. 105-112, 2010.

[4] D. Thilakanathan, S. Chen, S. Nepal and A. Rafael, "Secure Data Sharing in the Cloud," M. Pathan, S. Nepal, Security, Privacy and Trust in Cloud Systems, Springer-Verlag Berlin Heidelberg, 2014.

[5] A. Kahate, "Cryptography and Network Security," McGraw Hill, pp.38-198, 2016.

[6] Yogesh V Jilhawar, M Emmanuel,"Literature Survey on Different Cloud Computing Infrastructure-as-a-Service Frameworks", Networking and Communication Engineering Journal, vol 7, issue-1, pp 26-29, 2015.

[7] S. Wang, K. Liang, J. Liu, J. Chen, W. Xie and J. Yu, "Attribute-Based Data Sharing Scheme Revisited in Cloud Computing," IEEE Transactions on Information Forensics and Security, vol. 11, no. 8, pp. 1661-1673, 2016.

[8] J. Xiong, X. Liu, J. Ma, and G. Liu, "Ciphertext-policy hierarchical attribute-based encryption for fine-grained access control of encryption data," Int. J. Netw. Secur., vol. 16, no. 6, pp. 437–443, Nov. 2014.

[9] W. Susilo and K. Liang , "Searchable attribute-based mechanism with efficient data sharing for secure cloud storage," IEEE Trans. Inf. Forensics Security, vol. 10, no. 9, pp. 1981–1992, Sep. 2015.

[10] L. Cheung and C. Newport, "Provably secure ciphertext-policy ABE," in Proc. 14th ACM Conf. Comput. Commun. Secur., pp. 456–465, 2007.

[11] B. Waters and A. Sahai "Fuzzy identity-based encryption" Proc. 24th Annu. Int. Conf. Theory Appl. Cryptograph. Techn., pp. 457-473, 2005

[12] O. Pandey, V. Goyal, B. Waters, and A. Sahai. "Attribute-Based Encryption for Fine-grained Access Control of Encrypted Data," Proc. 13th ACM Conf. Computer and Comm. Security (CCS '06), pp. 89-98, 2006

[13] A. Sahai, B. Waters and J. Bethencourt "Ciphertext-policy attribute-based encryption" Proc. IEEE Symp. Secure. Privacy, pp. 321-334, May 2007

[14] S. S. M. Chow and M. Chase "Improving privacy and security in multi-authority attribute-based encryption" Proc. 16th ACM Conf. Comput. Commun. Secur., pp. 121-130, 2009.

[15] J. Hur "Improving security and efficiency in attribute-based data sharing" IEEE Trans. Knowl. Data Eng., vol. 25, no. 10, pp. 2271-2282, Oct. 2013

[16] X. Chen, X. Xie, H. Ma and J. Li "An efficient ciphertext-policy attribute-based access control towards revocation in cloud computing" J. Universal Comput. Sci., vol. 19, no. 16, pp. 2349-2367, Oct. 2013

[17] S. Katzenbeisser, S. Müller, and C. Eckert, "Distributed attribute-based encryption," in Proc. 11th Int.Conf. Inf. Secure Cryptol, pp.20– 36, 2009.

[18] C.-I. Fan, H.-M. Ruan and V. S.-M. Huang "Arbitrary-state attribute-based encryption with dynamic membership" IEEE Trans. Comput., vol. 63, no. 8, pp. 1951-1961, Aug. 2014

[19] J. Ma, X. Liu J. Xiong, Q. Li, and J. Ma, "Ciphertext-policy weighted attribute-based encryption for fine-grained access control," in Proc. 5th Int. Conf. Intell. Netw. Collaborative Syst., Sep., pp. 51–57, 2013.