International Journal of Electrical and Computer Engineering (IJECE)
Vol. 6, No. 3, June 2016, pp. 915 ~ 924
ISSN: 2088-8708, DOI: 10.11591/ijece.v6i3.8925

Cloud Security in Crypt Database Server Using Fine Grained Access Control

Krishna Keerthi Chennam[1], M. Akka Lakshmi[2]
[1] Department of Computer Science Engineering, Research Scholar, Gitam University, Hyderabad, India
[2] Department of Information Technology, HOD, Gitam University, Hyderabad, India

Article Info

Article history:

Received Aug 28, 2015
Revised Nov 24, 2015
Accepted Dec 8, 2015

Keyword:

CBE
CP-ABE
Fined Grained Access Control
Security

ABSTRACT

Information sharing in the cloud, powered by good patterns in cloud technology, is rising as a guaranteeing procedure for permitting users to advantageously access information. However, the growing number of enterprises and customers who stores their information in cloud servers is progressively challenging users' privacy and the security of information. This paper concentrates on providing a dependable and secure cloud information sharing services that permits users dynamic access to their information. In order to achieve this, we propose an effective, adaptable and flexible privacy preserving information policy with semantic security, by using Cipher text Policy Attribute Based Encryption (CP-ABE) consolidated with Character Based Encryption (CBE) systems. To ensure strong information sharing security, the policy succeeds in protecting the privacy of cloud users and supports efficient and secure dynamic operations, but not constrained to, file creation, user revocation. Security analysis demonstrates that the proposed policy is secure under the generic bi- linear group model in the random oracle model and enforces fine-grained access control, full collusion resistance and retrogressive secrecy. Furthermore, performance analysis and experimental results demonstrate that the overheads are as light as possible.

Corresponding Author:

Krishna Keerthi Chennam,
Department of Computer Science Engineering,
Gitam University,
Rudraram, Hyderabad, India
Email: krishnakeerthichl@gmail.com

1. INTRODUCTION

Cloud computing is currently developing as a technology in which cloud service providers (CSP) offer efficient data storage and computing facilities to a worldwide customers. The main necessity for a user is a connected terminal. By utilizing a combination of virtualization methods, service oriented computing, cloud computing, can be ordered into three types [1] X as an administration (XaaS), pay-as-you-go Services: the Platform as a Service (PaaS) model, e.g. Microsoft Azure (MIC), where users can send their own particular applications and tools to the cloud , Infrastructure as a Service (IaaS), e.g. Amazon EC2 (AMA), where clients can use cloud administrations given by the CSP to send self-assertive programming and Software as a Service (SaaS), e.g. Google App Engine (GAE), where clients use applications given by the CSS that run on the cloud foundation. Storing data in the cloud offers users the accommodation of access without requiring immediate knowledge of the deployment and administration of the hardware or infrastructure. In spite of the fact that cloud computing is substantially more effective than individualized computing, it brings new protection and security challenges, as users relinquish control by outsourcing their data they no longer having physical ownership of it.

Information holders request elevated amounts of security and confidentiality when they outsource the information to a cloud [2], although they generally encrypt their data while storing it in a cloud server, they still want control over it. Direct employment of traditional cryptographic primitives can't attain to the information security needed. Subsequently, a considerable amount of work has been controlled towards guaranteeing the protection and security of remotely shared data, utilizing a variety of systems and security models. These have mainly focused on preserving users' privacy while realizing desired security goals, without introducing excessively high levels of complexity to the users at the decryption stage. To illuminate these issues, scientists have either used Key- Policy Attribute Based Encryption (KP-ABE) [3] for secure access control or employed Hierarchical Attribute Based Encryption (HABE) [4] for information security. On the other hand, the HABE-based plan uses various leveled encryption to guarantee data security in a cloud, yet this presents too many private keys for each user to be managed effectively.

In summary, these plans either have security imperfections or provide security at the cost of execution. Therefore, the challenge of achieving the dual goals of privacy-preserving with effective cloud data sharing remains unresolved. To understand a powerful, adaptable and privacy preserving data imparting services in cloud computing, the following difficulties need to be met firstly, information holders ought to have the capacity to assign other cloud users with distinctive access privileges to their data, secondly, the cloud should have the capacity to help element demands so that information holders can add or revoke access privileges to different clients permitting them to make or erase their information, thirdly, the clients' security must be ensured against the cloud so that they can cover their private data while accessing the cloud and finally, users should have the capacity to impart information in the cloud through joined technologies with low compatibility, for example, smart phones and tablets.

A fine grained access control policies is proposed for privacy- preserving data sharing in the cloud that ensures both semantic security and effective availability of user data. To preserve privacy and guarantee data confidentiality against the cloud, employs a cryptographic primitive, named Cipher-text Policy Attribute -based Encryption (CP-ABE) [5] and combines it with a Character-Based Encryption (CBE) technique . Each data file is described by a set of meaningful elements, allowing each user to be assigned an access structure that defines the scope of data files they can have access. To protect user privacy, it does not need to update user secret key so that it prevents cloud users' access structure. To reduce the key management issue, the information holder basically assigns secret keys to users via the cloud.

The fundamental commitments can be compressed as follows:

1. We propose about the effective, scalable encryption for a cloud data sharing service that simultaneously achieve full privacy- preserving and data confidentiality.
2. The performance analysis incurs a small overhead compared to existing schemes,e, the experimental results demonstrate that the overheads are as light as possible.

2. RELATED WORK

An Attribute -Based Encryption (ABE) system was first proposed by Shahai and Waters [6]. It is essentially a simplified CBE system with only a single element. In an ABE scheme, the sender encrypts the message with a set of elements and specifies a number d; a recipient can only decrypt the encrypted message if they have at least d of the given Elements. Based on these principles proposed an ABE scheme with fine-grained data access control that supports monotonic access structures, such as AND, OR and other threshold gates. Ostrovsky etal [7] proposed an enhanced scheme that also supports non-monotonic access structures, i.e., NOT gates. Muller etal [8] presented a distributed Attribute based scheme, based on an efficient construction that demands a constant number of operations at the decryption stage. The access policy formats have to be expressed as a disjunctive normal form (DNF); therefore, the cipher text size is proportional to the number of conjunctive clauses in the DNF.

Chase [9] introduced a multi-authority ABE scheme in which several authorities cooperate to manage the Elements. Each authority manages a domain of Elements and distributes those Elements and secret keys to the users. The main issue affecting this scheme is that it is not practical to have one trusted central authority. An enhanced multi-authority ABE scheme was subsequently proposed by that remove the trusted authority, in order to preserve user privacy, each authority has to assign at least one Element to each user.

Lewko and Waters [10] provided a decentralized ABE scheme that does not require a trusted authority, but still maintains privacy. In their scheme, the access structure for any given user is only known by the sender. Decentralized mechanism is not suitable for cloud computing. The scheme proposed by Yu et al. [11] exploits KP-ABE, by combining it with proxy re- encryption and lazy re-encryption. It simultaneously achieves fine- graininess', scalability and data confidentiality for data access control. The information holder can delegate most of the computation tasks, such as user revocation, to the cloud server

without disclosing any data to the untrusted cloud, by delegating these tasks, some user attribute and secret keys may leak into the cloud. The related cipher text must be re-encrypted, allowing it to be revealed to non-revoked users. To discover the proxy re-encryption techniques applied in CP-ABE by Wang et al. [10],[11] provided a secure cloud storage scheme for health records in cloud computing by using Chase and Chow's multi-authority ABE scheme to divide users into different domains, this scheme is an isolated case and is not generally applied in cloud computing.

Vimercati [12] presents a formal access control model on outsourced data, where each file is encrypted with a symmetric key and each user is assigned a secret key, the complexity of operations of file creation and user grant/revocation is linear to the number of users, which makes this scheme unscalable. Moreover, Samarati [13] discusses some main privacy issues to be addressed in data outsourcing, ranging from data confidentiality to data utility, data protection and privacy over outsourced database scenarios. Wang et al. [11] proposed a hierarchical fine-grained access control scheme that relies on Hierarchical CBE and CP-ABE. The architecture of this scheme is arranged in a hierarchical way with a root master and several domain masters to generate keys for users, because a large number of keys are required for each entity, the system are complicated.

3. THEORITICAL WORK

3.1. CP–ABE Methodology

The framework model as indicated in Figure 1 Requires four parties in a system:

1. The information manager, who has information stored in the cloud and relies upon the cloud for information support.
2. Information holder can be enterprises or individual clients.
3. The information consumer, who goes through the information imparted by the information manager, downloads information of interest and decrypts it using his secret keys. The cloud server (CS) gives a high- quality service using various servers with significant storage space and computation power.
4. The secret key generator (SKG) is a trusted third party that processes by comparing private keys for users.

3.2. The Rival Model

The rival model considers most threats to cloud information confidentiality as malicious. Interestingly, the CS in the model is a semi-trusted (also known as passive), in that it acts appropriately majority of time, however, in specific circumstances an entity might subjectively deviate from the protocol specifications and the CS may attempt to secure as much secret information as possible. It proposes despite on the fact that the semi-trusted rival model is weaker than the malicious model; it is often a more realistic model. The three types of threats can be categorized as follows:

1. Internal threats (from the CSS and users who may acquire unapproved data), and external threats (from unapproved assaulters and outside foes beyond the system domain).
2. Active attacks (where unapproved users infuse malevolent files into the cloud), and passive attacks (where unapproved users eavesdrop on conversation between users and the cloud).
3. Collusion between the CSS and users (to access unapproved information with the end goal of harvesting file contents).

3.3. Security Requirements

As for secure information sharing and information access control in the cloud, the primary objective of proposed model is to protect the cloud information from being accessed by inner intruders, including the cloud and from external attackers and unapproved outside users.

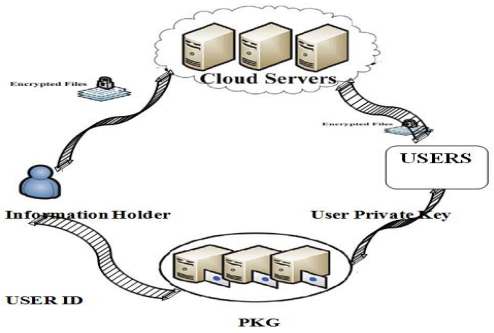


Figure 1. System Model

4. PROPOSED SCHEME

Framework has the following requirements:

1. In Fine- grained access control every user should only be able to access the data they are permitted to, with no access to unapproved information.
2. In Collusion safety, users should not be able to collude with some other user, or the cloud, with the end goal of sharing their secret key to access unapproved information.
3. In Retrogressive secrecy, the information access control approach should have the functionality to guarantee that users are not able to access the cloud information once their privileges have been revoked.

In order to improve privacy and security for data sharing in cloud computing, a scheme is proposed that combines CP-ABE and CBE. Figure 2 portrays a streamlined work process of the proposed plan.
In view of the system model, the proposed scheme is described in detail. The main objective of our proposed works is the authorized users to access and restore file effectively.

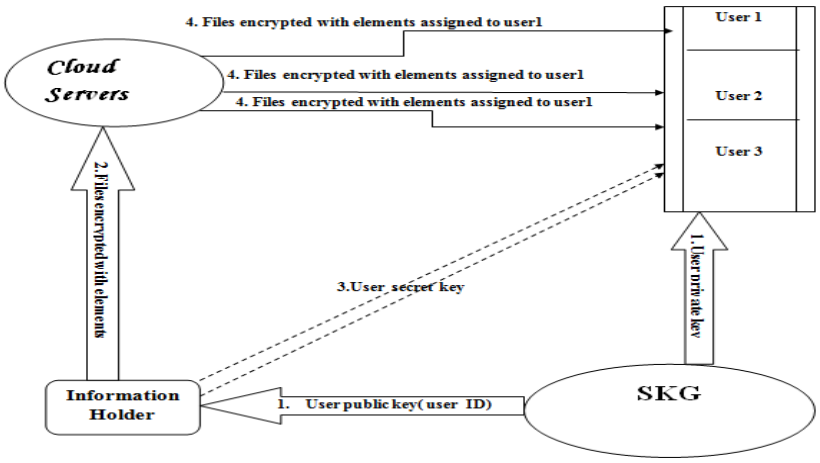


Figure 2. Simplified Workflow of Proposed Scheme

4.1. Access Policy

Access policy can be communicated with attributes at leaves and logic gates e.g. AND (^), OR (n) as intermediate node represented in ABE. Any access tree can be converted into the Boolean formula. Any access tree A can be converted over to a Linear Secret Sharing Scheme (LSSS) framework M. LSSS access structures are more general, and can be as resolved structure representations as Boolean equations. There are standard methods to change over any Boolean formula into a comparing LSSS grid. There are standard techniques to convert any Boolean formula into a corresponding LSSS matrix. The number of rows in the corresponding LSSS matrix will be same as the number of leaf nodes in the access tree. In LSSS, every piece is a vector over some finite field, and every set in the access structure reconstructs the secret using a linear combination of the coordinates of its pieces. Different from ABE, a message M is encrypted with an LSSS access structure (M, p) where p is a permutation function that maps rows of M to attributes in A. The user who only has the secret keys for a subset of rows Mx of M such that (1, 0, 0) is in the span of these rows can decrypt the message accurately.

4.1.1. Initialization

The information holder picks a large prime P, two groups G1, G2 of order P, and a map e: G1 ×G1→G2 and a hash function G1, G2 which maps a user ID to an element of G1. At that point the information holder characterizes a set of attributes A for sharing information records and chooses two arbitrary examples βi, γii Zp for each attributes in A. So the private key Prk for the system is

$$P_rK = \{\beta_i, \gamma_i, i \in A\}$$

The public key PuKof the system is published:

$$P_uK = \left\{e(g_1, g_1)^{\beta^i}, g_1^{\gamma^i}, i \in A\right\}$$

4.1.2. Encryption

The information holder characterizes a set of attributes I∈A for each data file. The configurations of access policy can be represented as n× l LSSS matrix M with a function ρ mapping its rows to attributes. The information holder processes the message M as follows:

- Randomly select a Zp, a random vector with the first entry as s. Let $\lambda_x = M_x.v$ where Mx is row x of M.
- Randomly select a vector $w \in Z_p^1$ with the first entry as 0 and a seed $r_x \in Z_p$. Let $w_x = M_x.w$
- encrypt the message M with (M, ρ) as follows:

$$c_{3,x} = g_1^{\alpha_{\rho(x)} r_x} g_1^{w_x}, c_{2,x} = g_1^{r_x}, c_{1,x} = e(g_1, g_2)^{\lambda_x} e(g_1, g_1)^{\beta_{\rho(x)} r_x}, c_0 = Enc_{e(g_1,g_1)^s}^{sym}(M) \forall x$$

Where p(x) is a permutation function mapping Mx to attribute, and $Enc_{e(g_1,g_1)^s}^{sym}(M)$ is a symmetric encryption under keys $e(g_1, g_2)$. Finally, the information holder uploads the encryption file $C = \{\forall x, \{C_0, C_{1,x}, C_{2,x}, C_{3,x}\}; (M, \rho)\}$ to the cloud servers.

4.1.3. Key Generation And Distribution

The information holder obtains user ID (IDu) from Secret Key Generator and assigns a set of attributes Iu for user Uu. Then the information holder calculates the private key component Pri, u for IDu of attribute i belonging to user U_u

$$Pr_{i,u} = g_2^{\beta_i} H(ID_u)^{r_i}$$

The private key for user Uu is $Pr_{i,u} = \{Pr_{i,u}, i \in I_u\}$. Pr(u) is encrypted by the user public key (IDu) and delivered to the user via the cloud server, such that only that user (IDu) can decrypt it using his private key.

4.1.4. Decryption

User Uu receives a cipher text $C = \{\forall x, \{C_0, C_{1,x}, C_{2,x}, C_{3,x}\}; (M, \rho)\}$ and H (IDu) from the cloud and selects constants such that $\Sigma_x c_x M_x = (1, 0, 0)$. The private key of Uu is $Pr_{i,u}, i \in I_u$. Then Uu calculates and obtain the message $M = D e_{e(g1,g1)^s}^{sym}(C_0)$. Using the proposed scheme, the information holder encrypts files and stores them into the cloud, while the users decrypt the cipher text C using their own secret keys.

4.2. Dynamic Operation

The information manager stores the information into the cloud server, where the cloud data dynamically changes. This is more applicable for some static application scenarios like libraries since the information holder stores the data into cloud server, rather than physically posing it, the dynamic data and user operations are quite challenging.

4.2.1. File creation

In cloud data sharing, there are cases when information holder uploads new data into the cloud servers. When the information holder wants to create a new file, he chooses a unique ID and defines the attribute set I for the new file. Then the information holder encrypts the file using the proposed algorithm and uploads the encrypted file and LSSS matrix with signature correctly to the cloud. If verifying the signature correctly, the cloud stores the new file. After uploading the encrypted file into the cloud, the information holder can go offline at any time.

4.2.2. File deletion

Sometimes, some antiquated cloud information ought to be erased. The delete operation considered here is straightforward. Just the information manager has the benefit to delete his stored file. At the point when the information manager needs to delete an antiquated file, he sends the file ID and his signature to the cloud. After verifying the signature on this file ID, the cloud deletes the antiquated file.

4.2.3. User operations

From users' perspective, to preserve the cloud data security, new users will join and outdated users need to be revoked.

4.2.4. User Addition

From the user's perspective, there are some new users who want to join the system to access the shared data. When a new user Uu joins the system, the information holder first obtains the user's ID (IDu) from the SKG, assigns the attribute set Iu and calculates the corresponding private key for this new user. The information holder then sends the private key and his signature to the cloud server. After verifying the signature, the cloud sends the secret key and related secret information to the new joining user. The user decrypts the message to get his secret key in the system. The information holder first obtains the new user ID (IDu) from PKG, assigns a set of attributes Iu for Uu and calculates the private key $P_{r(u)} = \{P_{r(i,u)}, i \in I_u\}$ for IDu. Then it encrypts the secret key, attribute set and the corresponding hash value H (IDu) $\left\{I_u, P_{r(u)}, H(ID_u), \delta_{0,(I_u, P_{r(u)}, H(ID_u))}\right\}$ with user's ID, denoting as D. Finally it sends cipher text D and user's ID $\left(D, ID_u, \delta_{0,(D,ID_u)}\right)$ to the cloud. After receiving the message from the information holder, the CS verifies the signature $\delta_{0,(D,ID_u)}$. If failed in signature verification, the CS deletes the received cipher text D to the joining user. The joining user first obtains his private key SK[u] from PKG. After decrypting the cipher text D using private Pr(u), he verifies the signature $\delta_{0,(I_u, P_{r(u)}, H(ID_u))}$.

Finally, the joining user accepts $I_u, P_{r(u)}, H(ID_u)$ as his access attribute set, secret key and user ID corresponding hash value. After receiving the secret keys, the newly joined user can access the matched files correctly. The cloud server only obtains the user's ID and system public key but no secret keys. Thus, privacy and security can be achieved.

4.2.5. User Revocation

At times, the information holder may renounce some users to gain access privileges. After being revoked, these users are not permitted to access the cloud information anymore. In some early works, the information manager overhauls the secret keys comparing to the attributes that the revoked user processes. At that point the information holder re- encrypts the related files and circulates the new keys to the non-renounced users by means of the cloud server. Despite the fact that it is additionally suitable for the proposed plan, it uncovers users' right to the cloud and brings more computation overhead. There is an optimizing method to deal with user revocation. The information holder just re-encrypts a piece of the cipher text and thus there is no need to update the corresponding private key. When there exists a user to be revoked, the information holder first determines the set of attributes Iu which user Uu possess. Then, he randomly chooses a new vector of. Now the new first entry of vector $(v)_{new}$ is $(s)_{new}$. A new $(\lambda_x)_{new} = M_x, (v)_{new}$ is calculated for each LSSS matrix row $(\lambda_x)_{new} = M_x, (v)_{new}$ x corresponding to attributes belong to Iu. The information holder re calculates the new values of $(C_{1,x})_{new}$ and as $(C_0)_{new}$.

$$(C_{1,x})_{new} = e(g_1, g_1)^{(\lambda x)_{new}} e(g_1, g_1)^{\beta_{0(x)} r_x}, (C_0)_{new} = Em_{e(g_1,g_1)^{(s)_{new}}(m)}$$

Finally he sends file ID f and user's ID along with the new encrypted file, $(f, ID_u, (C_0)_{new}, (C_{1,x})_{new}, \delta_0, (f, ID_u, (C_0)_{new}, (C_{1,x})_{new}$ to the cloud.

After verifying the signature $\delta_0, (f, ID_u, (C_0)_{new}, (C_{1,x})_{new}$ the CS deletes the old encrypted file and IDu from the UL. It stores the new received one on the base of file ID. Since we do not update the secret keys for non-revoked users, they access the cloud data just as given in Section 4.2. To prevent the revoked user eavesdrop the communication, the cloud can use non-revoked users' public key to encrypt the new encrypted file. In the stage of decryption, only the user obtains the exact C1, x can decrypt the message M, which can prevent the revoked user from accessing the cloud file.

4.3. Security Analysis

The security analysis focuses on the security requirements of the proposed scheme. In this scheme, assign flexible and different access privileges for each user to achieve fine grained access control. Meanwhile, the scheme achieves fully collusion secure which is important when several users collude and share their secret keys to access the unauthorized data. It also achieve user access privilege confidentiality.

4.3.1. Fine-grained access control

In this scheme, each user receives a flexible access structure from the information holder. Each user Uu has been assigned a set of attributes for the information owner. Suppose a file has an attribute i , so it has a corresponding row rb in the LSSS matrix. However, if the user Uu does not have the attribute i, he cannot receive the private key Pr (i, u) for attribute i. In addition, in the decryption stage as Uu cannot find the corresponding cx of row r to satisfy

$(C_{1,x})_{new} = e(g_1, g_1)^{(\alpha x)_{new}} e(g_1, g_1)^{\beta_{P(x)} r x}$, $(C_0)_{new} = En_{e(g_1,g_1)^{(y)}_{new}(m)}$ the decryption procedure will fail. Therefore, a user who does not have the attribute i cannot calculate $e(g_1, g_1)^{(s)}$. Thus, the user cannot decrypt the unauthorized message. Our scheme only discloses decryption keys to authorized users, thus unauthorized users and the cloud server cannot decrypt. But our proposed scheme can help the information holder to realize fine-grained access control of the cloud data.

4.3.2. Fully Collusion Secure

The scheme is fully collusion secure when users collude. Moreover, since the cloud and users do not have the private keys for unauthorized data, they are unaware of any information in regards to the unauthorized data, even if they collude each other. Therefore, scheme achieves fully collusion secure.

4.3.3. Retrogressive secrecy

Retrogressive secrecy can be realized in the proposed scheme. That is, the user who is revoked cannot decrypt the information which was previously able to be accessed. Scheme will update part of the cipher text C (1, x) after some legitimate users are revoked. Since C (1, x), which depends on the random s, is recalculated and not sent to the revoked user, the revoked user is not able to recover $e(g_1, g_1)^{(s)}$and decrypt the message. Therefore, the scheme is with retrogressive secrecy, the existing works, and the information needs to re-distribute keys for non-revoked users to guarantee retrogressive secrecy. The redistribution will disclose users' private key to the cloud and add additional communication cost.

4.3.4. User access privilege confidentiality

The proposed scheme does not disclose any attribute of a user attribute set to the cloud servers. In our key generation algorithm, users' access structures and private keys are assigned by the information holder. Thus, the cloud has no clue about users' private keys and does not possess any $P_{r\,(i,\,u)}$. Therefore, the cloud cannot derive any user's access privilege information so that users' privacy is protected against the cloud. Moreover, in user revocation and attribute change schemes, it's not necessary to update the non revoked users' secret keys. The cloud will transmit the new $C_{1,\,x}$ and C_0 to non-revoked users. According to the above analysis, see that the proposed scheme can achieve the desired security requirements, i.e., fine-grained access control, collusion resistance, and retrogressive secrecy. Furthermore, the information holder and user's identity is public in this scheme, but it is supposed to be hidden under some circumstances. Though it might increase some local computation; it does not significantly augment the overhead of computation and of communication.

5. RESULT AND DISCUSSION

In this section, the performance of proposed scheme is analyzed by comparing with other data sharing schemes that rely on CP ABE. We evaluate the computation and communication overhead, and our proposed system will give detail description about the cipher text size in the proposed scheme.

5.1. Computational complexity

Analyze the computation overhead of the proposed scheme according to the encryption and decryption algorithms in this section. In the proposed scheme, the main computation operations involved in encryption and decryption algorithms are pairing and scalar multiplication. Recall that the scheme chooses elliptic curve groups G1and G2 of order q. The cipher text of the proposed scheme is Y $C = \{\forall x, \{C_0, C_{1,x}, C_{2,x}, C_{3,x}\}\}$. Pairing is the most expensive operation. For each different file, however, information holder and users only need to calculate e (g1, g1) once in the beginning. Since both the proposed scheme and KP-ABE based schemes have the same numbers of pairing operation, do not involve in pairing operation overhead when computation complexity of the proposed scheme compares with the KP-ABE based schemes. In the computation complexity analysis, it takes account of scalar multiplication operation. During encrypting, all encryption operations are at the information holder side. The information holder needs to do two scalar multiplications to calculate C1, x, one scalar multiplication for $C_{2,x}(C_{2,x} = g_1^{r_x})$ and one for C3, x for each row in LSSS matrix. Therefore, the information holder needs at most $4(|A|)$ scalar multiplications.

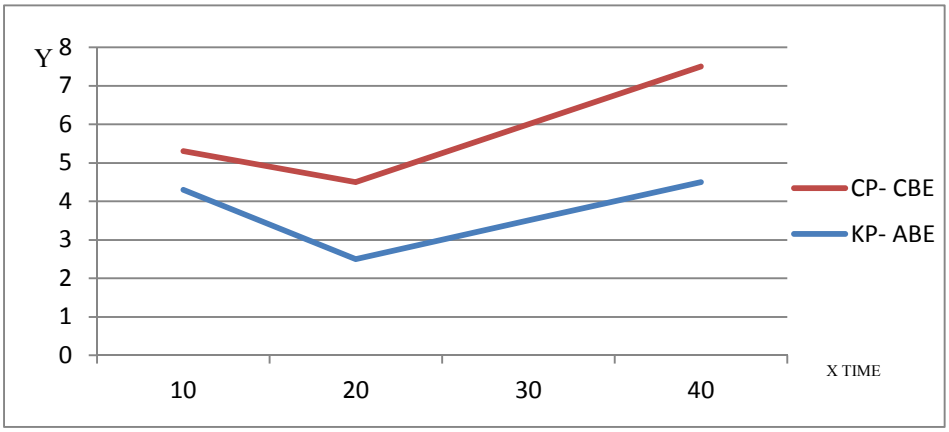


Figure 3. The overhead of key generation algorithms

The computation complexity of information holder convert the access structure to an LSSS matrix is $O(|I|)$ where $(|I|)$ the number of attributes about the access structure. Thus, the computation complexity of encryption is $O(|I|)$ in the decryption stage, the decryption operation is similar only for users. To recover cipher text, the user needs at most another $(|I|)$ scalar multiplications to calculate $\Pi_x\{e(g_1, g_1)^{\lambda x} e(H(ID), g_1)^{wx}\}$, so the time complexity is also $O(|I|)$. The computation complexity of CP-CBE and KP-ABE-based schemes is given in Table1.

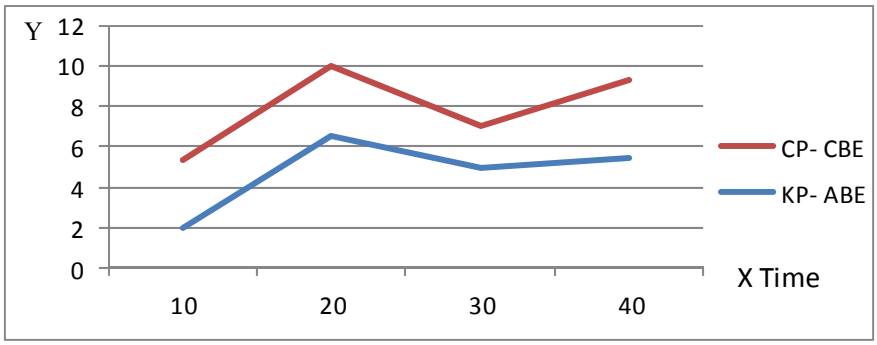


Figure 4. The overhead of encryption speed

Table 1. Computation complexity required in previous schemes and user access policy

Scheme	Encryption (Information holder)	Decryption (User)
KP-CBE	$O(max(\|I\|, N))$	$O(max(\|I\|, N))$
CP- CBE	$O(\|I\|)$	$O(\|I\|)$

5.2. Revocation Cost

When user revocation is required, the cipher text needs to be re-encrypted in our scheme. The information holder will choose a new seed s randomly and recalculate C0 and C1, x. suppose the revoked user is Uu. Pairing (e (g1, g1)) has been calculated, so the information holder only needs one scalar multiplication to recalculate C0. For each attribute x $2|I_u| + 1$ Iu, there are another two scalar multiplications to update C1, x. Therefore, there are totally $2|I_u| + 1$ scalar multiplications to re-encrypt the cipher text by the information holder. For the non-revoked users, they do not need to do any computation. Moreover, the information holder needs to send the new cipher text to the cloud, while the cloud just replaces the outdated cipher text and does not need to transfer it to the non-revoked users, so the additional communication costs is

$(|U| + 1)\log|G_1| +$ Data. Compared with proposed scheme, this brings an abundance of additional computation and communication overhead. The proposed scheme can accomplish this dynamic request with lightweight computation complexity.

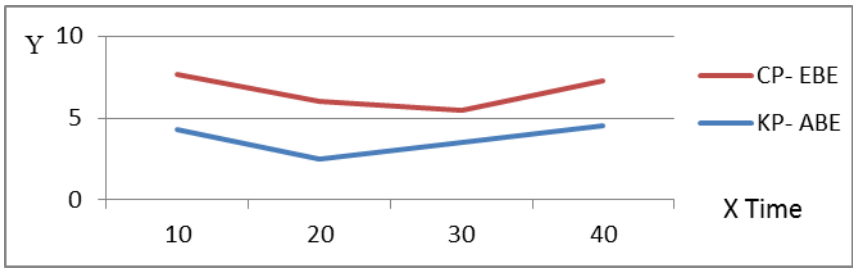


Figure 5. The overhead of decryption algorithm

6. CONCLUSION

In this paper, a privacy- preserving and secure information sharing scheme in cloud computing by exploiting CP- ABE and consolidating it with method of CBE is displayed. The proposed scheme guarantees fine-grained data access control, retrogressive secrecy and security against collusion of users with the cloud and supports client expansion, denial and characteristic alterations. Besides, proposed scheme does not unveil any elements of users to the cloud so that it keeps the privacy of the users away from the cloud. Security analysis demonstrates that the proposed scheme is semantically secured in the non specific bilinear gathering model, modeling H as a random oracle. Likewise, access the execution of the proposed scheme about computation complexity. The result demonstrates that the proposed scheme is low overhead and highly efficient. Emulating the flow research, it will implement the proposed privacy- preserving and effective cloud information sharing service in a real CSP platform for future work.

ACKNOWLEDGEMENTS

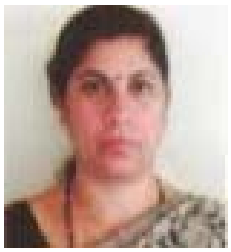
I thank to my Guide and the Organization who supported me to publish my work in your journal.

REFERENCES

[1] M. Armbrust, *et al.*, "Above the clouds: a Berkeley view of cloud computing [Technical report]," Berkeley, EECS Department, University of California, 2009.
[2] Erway C., *et al.*, "Dynamic provable data possession," in *Proceedings of the 16th ACM conference on computer and communications security (CCS)*, ACM, 2006, pp. 213e22.
[3] Goyal V., *et al.*, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on computer and communications security (CCS)*, 2006.
[4] Boneh D., *et al.*, "Hierarchical identity based encryption with constant size cipher text," in *Advances in Cryptology*, eEUROCRYPT, Springer, 2009.
[5] Bettencourt J., *et al.*, "Ciphertext-policy attribute based encryption," in *IEEE Symposium on security and privacy (SP)*, IEEE, 2007, pp. 321e34.
[6] Sahai A., *et al*., "Fuzzy identity-based encryption. In: Advances in cryptology," in *EUROCRYPT*, Springer, 2005, pp. 557e73.
[7] Ostrovsky R., "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 14th ACM conference on computer and communications security (CCS)*, 2007, pp. 195e203.
[8] Mu¨ller S., "Distributed attribute-based encryption," in *Information security and cryptology (ICISC)*, Springer, 2009.
[9] Chase M., "Multi-authority attribute based encryption," in *Theory of cryptography (TCC)*, Springer, pp. 515e34, 2007.
[10] Lewko A. and Waters B., "Decentralizing attribute-based encryption," in *Advances in Cryptology EUROCRYPT*, Springer, 2011, pp. 568e88.
[11] Yu S., *et al.*, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *International conference on computer communications (INFOCOM)*, IEEE, 2010, pp. 1e9.
[12] S. D. C. di Vimercati, *et al.*, "Over-encryption: management of access control evolution on outsourced data," in *ACM*, 2007.
[13] Samarati P. and De Capitani di Vimercati S., "Data protection in outsourcing scenarios: issues and directions," in *Proceedings of the 5th ACM Symposium on information, computer and communications security (ASIACCS)*, ACM, 2010, pp. 1e14.

BIOGRAPHIES OF AUTHORS

Krishna Keerthi Chennam obtained Bachelor's degree in computers science engineering from JNTU, Hyderabad in 2005, received the Masters Degree in Embedded Systems from JNTUH, in 2012 and pursuing PhD in CSE from Gitam University, Hyderabad campus. Research interests include Cloud Computing, Cloud security. Currently working as Assistant Professor in Computer Science & Engineering Department at Muffakham Jah College of Engineering & Technology, Banjarahills, and Hyderabad.



Dr.M.Akkalakshmi received PhD from Osmania University in 2008. Her research focus in Network Security, Cloud Computing, Cloud Security, Big Data. She is presently working as Professor and IT-HOD in Gitam University, Hyderabad Campus. She is the author of several research papers in the area of Network Security and Cloud computing.