Received: June 13, 2017

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Cloud Data Integrity Auditing Over Dynamic Data for Multiple Users

Santhosh Kumar[1*] **Latha Parthiban[2]**

[1]*Sathyabama University, Tamilnadu, India*
[2]*Pondicherry Community College, Pondicherry, India*
* Corresponding author's Email: santhoshcloudcom@gmail.com

Abstract: Cloud computing is a state-of-the-art computing model, which encourages remote data storage. This facility shoots up the necessity of secure data auditing mechanism over outsourced data. Several mechanisms are proposed in the literature for supporting dynamic data. However, most of the existing schemes lack the security feature, which can withstand collusion attacks between the cloud server and the abrogated users. This paper presents a technique to overthrow the collusion attacks and the data auditing mechanism is achieved by means of vector commitment and backward unlinkable verifier local abrogation group signature. The proposed work allows multiple users to deal with the remote cloud data. The performance of the proposed approach is checked in terms of update, verify and enquiry time cost. The performance of the proposed work is analysed and compared with the existing techniques. The update time cost is lesser than the comparative techniques. However the verify time cost is greater, because of the process of integrity verification of the signature. Hence, this work ensures quality of service and tightened security by having reasonable update time and a strict verification policy respectively.

Keywords: Cloud computing, Data auditing, Vector commitment.

1. Introduction

Cloud computing is a new generation computing model that provides an option to distribute the computational and storage resources. It provides a range of Information Technology (IT) services to the users via network with high computational and storage ability at low charge [1]. Cloud storage services are the boon to the small and mid-scale industry, as it involves a nominal charge based on demand. The cloud storage provides a smart solution to issues as management and maintenance of data. Besides these advantages, the major concern about cloud is the security of data.

Though the cloud service provider guarantees the security of data, the data owners are still reluctant for remote data storage, as the data may get corrupted. The data integrity may get affected either intentionally or unintentionally. Thus, data integrity preservation is a challenging task and several mechanisms have been presented to handle this issue. For instance, a secure and reliable cloud storage service based on Luby Transform (LT) codes is presented in [2]. In [3], a privacy preserving public auditing mechanism is proposed namely Oruta. The verification information of this mechanism is computed by ring signatures. A dynamic auditing service for integrity verification is proposed in [4]. The auditing service of this work is based on fragmentation, random sampling and index hash table. Initially, the integrity solutions were simple replications of stored data, which is quite impractical now-a-days due to the exponential data growth.

The data integrity solutions can be categorized into two and they are static and dynamic. The static data integrity solution does not accept any sort of data modification. However, the integrity solution that supports data dynamism is necessary. A data integrity scheme is claimed as public verifiable, when the third party auditor checks for the integrity, in addition to the data owner. Most of the existing dynamic schemes for integrity verification assume that the data owner alone can perform data modification. For instance, the work proposed in [5] achieves Proof of Retrievability (PoR) by means of hardness amplification. In [6], a dynamic provable data possession model, which is an enhancement of provable data possession, is proposed.

A privacy preserving public auditing system is proposed in [7], which employs homomorphic linear authentication and random masking techniques, in order to prevent data leakage. A PoR scheme is proposed in [8], which is based on constant size polynomial commitment and homomorphic linear authenticators. In [9], a dynamic PoR, which is public verifiable is proposed. Several variants of PoR schemes such as bounded and unbounded usage is presented in [10]. However, all these works prompts the data owner to perform data modification.

Of late, cloud is mainly utilized for data sharing and thereby encourages group effort. In such platform, the cloud customers of a group share and access the data from any cloud participating in a group. The existing solutions assume the data modification is done by data owner alone. Recognizing the importance of this issue, this work proposes to present a reliable integrity verification model along with effective group user abrogation. The contribution of the proposed work is listed below.

- A defended and effective data integrity auditing scheme is presented for user group for cipher text database.
- An effective data auditing scheme is proposed by incorporating polynomial commitment, dynamic group key agreement protocol and unlinkable group signature.

The remainder of this paper is organized as follows. Section 2 presents the related works with respect to data integrity. Section 3 and section 4 present the preliminaries and background respectively. The proposed work is presented in section 5. The performance of the proposed work is evaluated in section 6. The concluding remarks are drawn in section 7.

2. Related works

The intention of this section is to review the existing literature with respect to data integrity preservation in cloud environment.

The theory of Provable Data Possession (PDP) and Proofs of Retrievability (PoR) were initially introduced by Ateneise et.al. and Juels et.al. respectively [11,12]. As these works are initial versions, they could not provide advanced functionality and efficiency. In [13], a verifiable database scheme on the basis of subgroup membership problem in bilinear group is presented. Yet, the public verifiability attribute is not considered. The concept of group signature is introduced by Chaum and Heyst [14]. The group signature promotes the concept of anonymous signers, such that every group member possesses a private key for signing messages. Besides this, the identity of the signer is kept secret. An efficient group signature with verifier-local revocation, which supports traceability and anonymity, is proposed in [15]. However, this involves greater communication and computation overheads. Another group signature method is presented in [16], which is based on broadcast encryption leads to memory overhead. An enhancement of this work is presented in [17], which tends to provide static sized private keys. Yet, the the issue of memory overhead could not be solved completely.

A data integrity method on the basis of ring signature is proposed, in order to support data operations by several users of a group [3]. However, this scheme does not take the issue of user revocation into account and the cost of auditing is directly proportional to the size of group and data.

The enhancement of this scheme is presented in [18], which employs proxy re-signatures. The drawback of this scheme is the assumption of the presence of secure channel between the entities and the collusion is not taken into account. Besides this, the auditing cost is directly proportional to the size of the group or team. In [19], a dynamic public integrity auditing scheme is presented and is proved to be efficient with user revocation. The demerit of this scheme is its working nature with plain text but not cipher text.

Motivated by the above stated works, the proposed scheme intends to work effectively with the cipher text database. Thus, taking all the aforementioned points into account, the proposed work intends to overcome this issue by means of leveraging dynamic asymmetric group key protocol [20] and backward unlinkable group signature [21]. The asymmetric group key protocol permits the group members to establish a dynamic public group encryption key and each member possesses a different decryption key.

3. Preliminaries

The intention of this section is to explain the storage representation of the proposed work and to present the threat model along with the security goals to be attained.

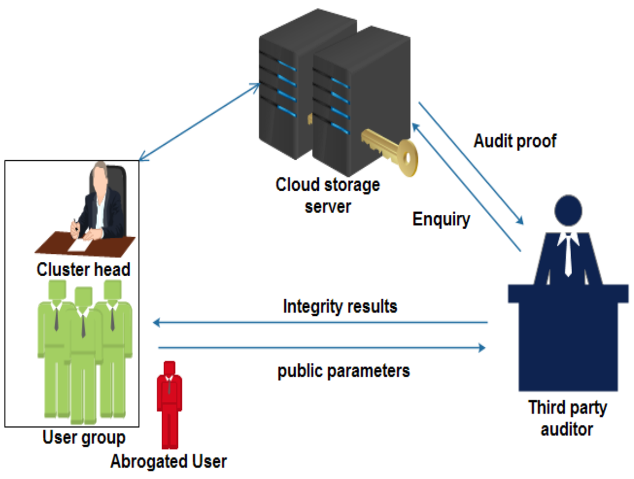

Figure.1 Data storage in cloud

3.1 Representation of cloud storage

The cloud storage pattern of the proposed work relies on three key entities. They are cloud server, clustered users and Third Party Checker (TPC). A team of users is named as clustered users, which contains a data owner or cluster head and a group of users. The users can access or modify the data which is subjected to the cluster head permission. Cloud server is utilized for data storage and is not considered trustworthy. TPC is responsible for the data integrity verification of the data being saved in the cloud server. The overall idea of data integrity auditing is presented in figure 1.

The cluster head encrypts the data to be uploaded and stores it in the cloud server. Besides this, the cluster head is the authority to grant permission to the cluster users and to revoke the privilege assigned to the user. The TPA checks for the data integrity and serves its best even when the data is modified often.

3.2 Security threats and design goals

The clustered user may be abrogated by the head of the cluster, at any instant of time with respect to the behaviour. In such cases, the abrogated user may perform fraudulent activity with the cloud server and share the cluster's secret key. This is a serious issue as the secret key of the abrogated user is obtained by the partially trusted cloud server. This work strives to overcome the aforementioned issues by introducing several security measures.

4. Background

The proposed technique exploits bilinear groups. The technique's security is based on Decision Linear assumption, ℓ-Hidden Strong Diffie-Hellman assumption and Decision Tripartite Diffie-Hellman assumption. This section aims to review the definitions and complexity assumptions of the system.

4.1 Bilinear groups

Consider $\mathcal{g}$ and $\mathcal{g}_T$ are the groups of prime order P. These groups are called as bilinear groups and their mapping would be $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, such that

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

$$for\ any\ (g,h) \in \mathbb{G} \times \mathbb{G} \text{ and } a, b \in \mathbb{Z}; \quad (1)$$

$$e(g,h) \neq 1_{\mathbb{G}_T}\ when\ g,h \neq 1_{\mathbb{G}} \quad (2)$$

These groups involve non-interactive complexity assumptions and are given below.

Def.1. Decision linear problem

Let $\mathcal{g} = < g >$ be a group which of prime order $P > 2^{\lambda}$. The purpose of decision linear issue is to differentiate the distributions $(g, g^a, g^b, g^{ac}, g^{bd}, g^{c+d})$ and $(g, g^a, g^b, g^{ac}, g^{bd}, g^{z})$, where $a, b, c, d \overset{R}{\leftarrow} \mathbb{Z}_p^*$, $z \overset{R}{\leftarrow} \mathbb{Z}_p^*$. The decision linear problem can make the decision about the linearity of the vectors, in order to predict the dependency. Thus, it is more suitable for establishing non-interactive proof systems.

Def.2. ℓ-Hidden Strong Diffie-Hellman problem

Let $\mathcal{g}$ be a group of order p. The ℓ-Hidden Strong Diffie-Hellman problem with factors $(g, \Omega = g^{\omega}, u) \overset{R}{\leftarrow} \mathbb{G}^3$ and ℓ distinct triples $(g^{\frac{1}{\omega+s_i}}, g^{s_i}, u^{s_i})$ with $s_1, \ldots s_l \overset{R}{\leftarrow} \mathbb{Z}_p^*$ in order to compute another triple $(g^{\frac{1}{\omega+s}}, g^s, u^s)$, so $s \neq s_i\ for\ i \in \{1,2,..l\}$.

Def.3. Decision Tripartite Diffie-Hellman problem

Let $\mathcal{g}$ be a group of order p. The Decision Tripartite Diffie-Hellman problem checks for the infeasibility if $\eta = g^{abc}$ over (g, g^a, g^b, g^c, η) as the input; where $a, b, c \overset{R}{\leftarrow} \mathbb{Z}_p^*$. This assumption is quite tougher than Decision Bilinear Diffie-Hellman assumption.

4.2 Vector commitment

Commitment techniques are the vital components of cryptographic algorithms and some of the key properties of it are vote, identification and zero-knowledge proof etc. A secure commitment scheme works by having an entity called 'committer', who is permitted to publish a value. This process binds the committer with the message, however maintains secrecy. The committer can open the commitment later on, in order to disclose the committed message to the verifier. The verifier then checks for the data consistency along with the commitment.

The vector commitment scheme is proposed in [22]. The main features of vector commitment scheme are listed below. The vector commitment supports position binding, such that the attacker cannot open the commitment with two distinct values from the constant location. Besides this, vector commitments do not rely on the vector length.

Def.4. A vector commitment scheme is comprised of six different algorithms which are $VC.KeyGen, VC.Com, VC.Open, VC.Ver,$ $VC.Update, VC.ProofUpdate$.

$VC.KeyGen(1^k, q)$ - Given the security parameter k and the size q of the committed vector (with $q = poly(k)$), the key generation outputs some public parameters pp (which implicitly defines the message space M).

$VC.Com_{pp}(m_1, \ldots m_q)$ - On input a sequence of q messages $m1, \ldots, mq \in M$ and the public parameters pp, the committing algorithm outputs a commitment string C and an auxiliary information aux.

$VC.Open_{pp}(m, i, aux)$ - This algorithm is run by the committer to produce a proof Λi that m is the i^{th} committed message. In particular, notice that in the case when some updates have occurred the auxiliary information aux can include the update information produced by these updates.

$VC.Ver_{pp}(C, m, i, \Lambda i)$ - The verification algorithm accepts (i.e., it outputs 1) only if Λi is a valid proof that C was created to a sequence $m_1, \ldots, m_q$ such that $m = m_i$.

$VC.Update_{pp}(C, m, m', i)$ - This algorithm is run by the committer who produced C and wants to update it by changing the i^{th} message to m'. The algorithm takes as input the old message m, the new message m' and the position i. It outputs a new commitment C'together with a update information U.

$VC.ProofUpdate_{pp}(C, \Lambda_j, m', i, U)$ - This algorithm can be run by any user who holds a proof Λj for some message at position j with respect to C, and it allows the user to compute an updated proof Λ'_j (and the updated commitment C') such that Λ'_j will be valid with respect to C'which contains m' as the new message at position i. Basically, the value U contains the update information which is needed to compute such values [22].

Recently, the works proposed in [23,24] claimed that the vector commitment scheme is susceptible to forward automatic update attack and backward substitution attack. The solution for the same is also presented in those works.

4.3 Unlinkable group signature

The formal definition of unlinkable group signature is presented below. The backward unlinkable verifier local revocation is comprised of the following algorithms.

$KeyGen(\lambda, N, T)$ - This randomized algorithm taking as input a security parameter $\lambda \in \mathbb{N}$ and integers $N, T \in \mathbb{N}$ indicating the number of group members and the number of time periods, respectively. Its output consists of a group public key gpk, a N-vector of group members' secret keys $gsk = (gsk[1], \ldots gsk[N])$ and a $(N \times T)$ vector of revocation tokens $grt = (grt[1][1], \ldots, grt[N][T])$, where $grt[i][j]$ indicates the token of member i at time interval j.

$Sign(gpk, gsk[i], j, M)$ – It is a possibly randomized algorithm taking as input, the group public key gpk, the current time interval j, a group member's secret key $gsk[i]$ and a message $M \in \{0,1\}^*$. It outputs a group signature σ.

$Verify(gpk, j, RL_j, \sigma, M)$ – It is a deterministic algorithm that takes gpk as input, the period number j, a set of revocation tokens RL_j for period j, a signature σ, and the message M. It outputs either "valid" or "invalid". The former output indicates that σ is a correct signature on M at interval j with respect to gpk, and that the signer is not revoked at interval j. The main features of unlinkable group signature are traceability and anonymity [25].

5. Proposed scheme

The proposed work takes a database Dbase into account, which consists of multiple records (id, val_{id}), where id is the index and val_{id} is the value of that index. The proposed scheme supports dynamism and so the stored data can be accessed and modified by the user. Finally, the data integrity can be verified. The building blocks of proposed scheme are given below.

$\boldsymbol{Setup(1^k, Dbase)}$

Consider a Dbase with (id, val_{id}), where $1 \le id \le w$. The database is maintained by clustered users with a cluster head. The cluster head is responsible for granting permission or revoke the granted permission from the users.

1. Initially, the KeyGen algorithm of vector commitment is executed, so as to obtain the public parameters (pp), which can be denoted as $pp \leftarrow VC.KeyGen(1^k, w)$.
2. Execute the KeyGen of backward unlinkable VLR group signature, in order to acquire group public key (gpk), secret key of group members (gsk) and revocation tokens (grt). This can be denoted by $(gpk, gsk, grt) \leftarrow VLR.KeyGen(1^k, N, T)$, where $gsk = (gsk[1], gsk[2], .. gsk[n])$ and $grt = grt[1][1], \dots, grt[N][T]$. N is the count of group members and T is the time interval.
3. Compute the commitment and supplementary information by $(C, aux) \leftarrow VC.Com_{pp}(c_1, \dots, c_w)$. Consider cur_{usr} as the current data updater, such that $0 \leq cur_{usr} \leq N$ and assume that $(gsk[cur_{usr}], gpk)$ be the secret and public key of the corresponding clustered user. Let the commitment be denoted as $C^t = VC.Com_{pp}(c_1^t, .. c_w^t)$, where t is the counter.
4. Execute the signing algorithm upon the commitment C. The signature is computed by taking $gpk, gsk[cur_{usr}] and\, C$ into account. The current user cur_{usr} computes the signature at the specific time interval, which can be given by $\sigma^t \leftarrow VLR.Sign(gpk, gsk[cur_{usr}], t, C)$. The so computed signature σ^t is forwarded to the cloud server. The cloud server checks for the validity of σ^t and computes $C(t) = \sigma^t \times C^t$. This supplementary information is added to the aux. query
5. Set the public key

$$PK = (pp, gpk, C(t-1), C(t)) \qquad (3)$$

$\boldsymbol{Enquiry(PK, pp, aux, Dbase, k)}$

In this stage, the clustered user has to execute the open algorithm, in order to produce the proof $\Lambda_k \leftarrow VC.Open_{pp}(c_k, k, aux)$; Λ_k is the proof of k^{th} committed message. The outcome of this phase is

$$\rho = (c_k, \Lambda_k, \sum(t)) \qquad (4)$$

$\boldsymbol{Verify(PK, j, RL_j, \sigma, C)}$

Once the proof is proved to be valid, then the verification algorithm of group signature is executed. $VLR.Verify(gpk, j, RL_j, \sigma, C)$ takes the public key, time period, revocation tokens at time period j, signature and the string. σ is determined by $sign(gpk, gsk[n], j, C)$. The verification algorithm determines the validity. This is followed by the execution of verification algorithm of vector commitment.

$\boldsymbol{Update(k, \rho)}$

In the update phase, initially the user enquiries and verifies the database for checking the validity, by following the previously explained sessions. The update operation can be carried out by taking the index, message and new message into account. The outcome of this operation is a new commitment string along with the updated message, which can be represented by the following.

$$(C', U_m \leftarrow VC.Update \\ (C, old\ message, new\ message, i) \qquad (5)$$

$\boldsymbol{ProofUpdate}$

The user who holds the proof Λ_k for the message at k^{th} position for the committed string C can proceed with this phase. The updated proof Λ'_k can then be created for the committed string C' with the newly updated message at position k.

$\boldsymbol{User\ abrogation}$

The process of user abrogation can be accomplished by the third party auditor by executing the verification algorithm of the backward unlinkable VLR. As a clustered user group contains multiple users, it is essential to trace the user who produces the signature by utilizing the grt, as the trace key. The signed user can be verified by the message opener by running the verification algorithm which takes the message and the signature for a specific time interval as input. The verification algorithm is applied over the message along with the signature by exploiting the revocation tokens $RL_j = \{grt[m][n]\}\ m \in 1 \dots N$; N is the count of clustered user group. The corresponding index is returned as the output and the status is set either as valid or invalid.

5.1 Support for encrypted database

Usually, the data owners prefer to encrypt the database before the process of outsourcing data. Thus, it is obviously necessary for the auditing mechanism to support the encrypted database. Encryption is the process of changing the original data into unintelligible format. In the cloud environment, a single user can achieve data encryption effectively. For instance, the data d_x can be encrypted by any encryption mechanism with a secret key. The encryption process of a single user environment is hassle-free. However, a single secret key for a user group does not serve the purpose for encryption in a multiple user environment. The usage of single secret key can introduce several issues such as single point-of-failure and security breaches.

For this sake, a mechanism which can support data alteration in a user group is needed. In [24], group key agreement protocol namely Dynamic Identity-based Authenticated Asymetric Group Key Agreement (IBAAGKA) is presented. This technique allows the user group to create a public group encryption key, in a dynamic fashion and each user has unique key. This scheme is proved to be secure against k -bilinear Diffie-Hellman exponent assumption.

The proposed work utilizes the aforementioned key agreement protocol, in order to extend its support to encrypted database. The changes incorporated in the proposed work are listed below. The key agreement protocol has to be executed in the setup phase. Let (id, val_{id}) be the database with id as indexes and val_{id} is the value. This database is encrypted by the group public key of the cluster head, in order to arrive at a database $(cid, cval_{id})$.

In the process of updation, the clustered user is prompted to decrypt the data with the help of the secret key $gsk[i]$, in order to arrive at the original database. This step is followed by altering the data to val'_{id}. The so updated data is again encrypted by means of the group public key, such that $(cid', cval'_{id})$ is attained.

6. Performance analysis

The performance of the proposed work is compared with the results of [25,26]. All the mentioned works need an expensive set up phase. The enquiry cost of the proposed work is directly proportional to the count of data items. However, the computational cost of the proposed work is comparatively low. The enquiry cost of the proposed work is presented in figure 2. From the experimental results, it can be observed that the time cost grows along with the data items.

The verify time cost of the proposed work is analysed and compared with the existing schemes in figure 3. It is observed that the verify time cost of the proposed work is five times greater than the existing works. The reason for the increase in 'verify time cost' is that the verification phase involves the integrity verification of signature, which involves several parameters.

On analysing the update time cost as shown in figure 4, it is evident that the update time cost of

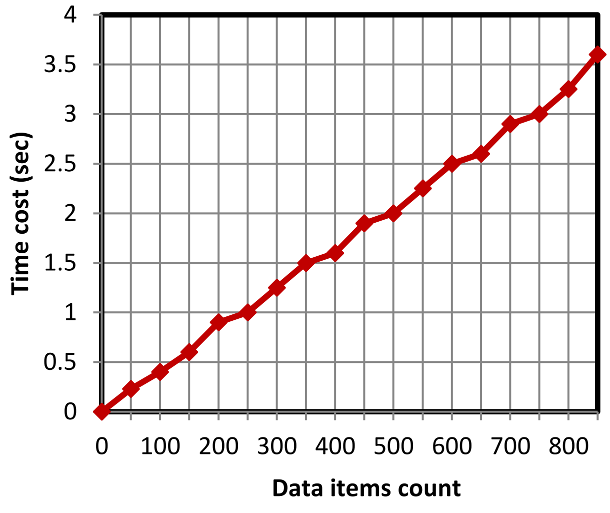

Figure.2 Enquiry time cost

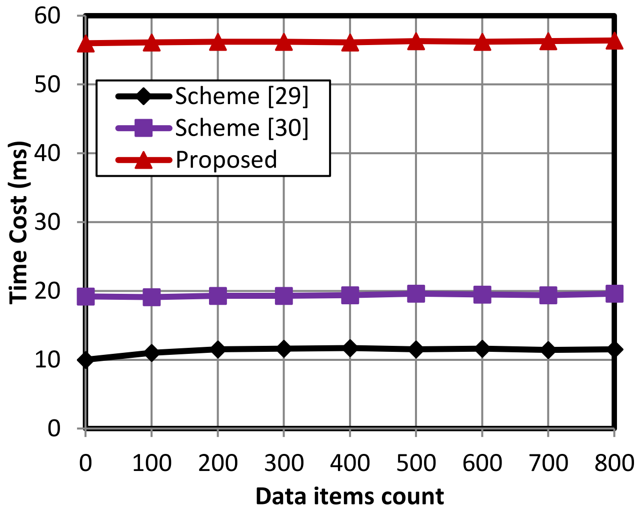

Figure.3 Verify time cost

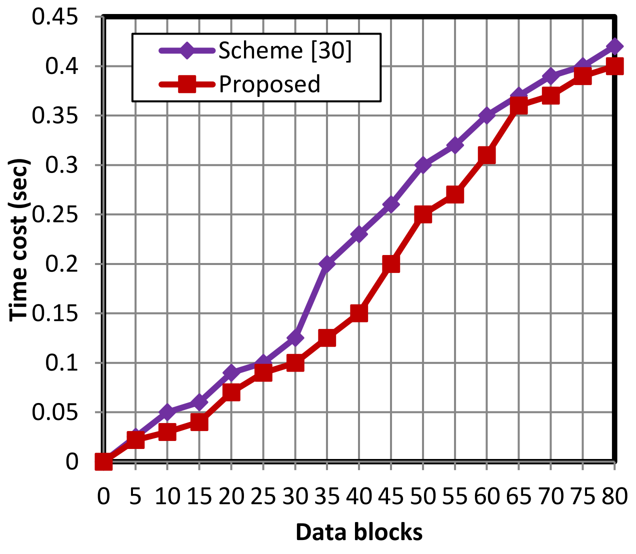

Figure.4 Update time cost

both the techniques show gradual rise with respect to the data blocks. However, the update time cost of the proposed approach is considerably low, as the data updation can be done in a stretch upon data owner's approval. The experimental analysis evaluates the time cost of enquiry, verify and update operations, owing to their importance. From the experimental analysis, it is proven that the update time cost of the proposed work is lower than the existing work. Thus, the proposed work supports user groups, works over cipher text database and proves high degree of security.

7. Conclusion

The cloud storage must guarantee effortless data modification to the cloud users. This paper presents a scheme that operates over dynamic data, which provides secure data integrity auditing mechanism for user group with the alter provision. The data integrity auditing is achieved by exploiting vector commitment, Dynamic Identity-based Authenticated Asymetric Group Key Agreement (IBAAGKA) and backward unlinkable verifier local revocation group signature. These ingredients make it possible to achieve working with cipher text database and secure user abrogation. Besides this, the proposed work is resistant against collusion attacks.

The proposed work supports all sorts of dynamic data operations and frees the data owners from the worry of data modification. This work paves for the cloud customers of a group can share and access the data. The data owner has all the rights to abrogate the user at any instant of time. This ensures data security, while providing good quality of service. The experimental results of the proposed work are satisfactory in terms of computational and time complexity. In future, this research work can be improved by reducing the verification time cost.

References

[1] L.M. Vaquero, L.R. Merino, J. Caceres and M. Lindner, "A break in the clouds: towards a cloud definition", *ACM SIGCOMM Computer Communication Review*, Vol.39, No.1, pp.50-55, 2008.

[2] Y. Zhu, H. Wang, Z. Hu, G. J. Ahn, H. Hu, H. and S.S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds", In: *Proc. of ACM Symposium on Applied Computing*, Taichung, Taiwan, pp. 1550-1557, 2011.

[3] B. Wang, B. Li, and H. Li, "Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud", *IEEE Transactions on Cloud Computing*, Vol.2, No.1, pp. 43-56, 2014.

[4] N. Cao, S. Yu, Z. Yang, W. Lou, and Y. T. Hou, "LT Codes-based Secure and Reliable Cloud Storage Service", In: *Proc. of IEEE INFOCOM*, Florida, USA, pp. 693–701, 2012.

[5] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Proofs of retrievability via hardness amplification", In: *Proc. of ESORICS*, Saint-Malo, France, pp. 355–370, 2009.

[6] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession", In: *Proc. of ACM CCS*, Illinois, USA, pp. 213–222, 2009.

[7] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing", In: *Proc. of IEEE INFOCOM*, CA, USA, pp. 525–533, 2010.

[8] J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud", In: *Proc. of International Workshop on Security in Cloud Computing*, Hangzhou, China, pp. 19–26, 2013.

[9] E. Shi, E. Stefanov, and C. Papamanthou, "Practical dynamicproofs of retrievability", In: *Proc. of ACM CCS*, Berlin, Germany, pp. 325–336, 2013.

[10] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification", In: *Proc. of TCC*, CA, USA, pp. 109–127, 2009.

[11] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores", In: *Proc. of ACM CCS*, pp. 598–610, 2007.

[12] A. Juels and B. S. Kaliski, "Pors: Proofs of retrievability for large files", In: *Proc. of ACM CCS*, Virginia, USA, pp. 584–597, 2007.

[13] S. Benabbas, R. Gennaro, and Y. Vahlis, "Verifiable delegation of computation over large datasets", In: *Proc. of CRYPTO*, CA, USA, pp. 111–131, 2011.

[14] D. Chaum and E.V. Heyst, "Group signatures", In: *Proc. of EUROCRYPT*, Brighton, UK, pp. 257–265, 1991.

[15] D. Boneh and H. Shacham, "Group signatures with verifier local revocation", In: *Proc. of ACM CCS*, DC, USA, pp. 168–177, 2004.

[16] B. Libert, T. Peters, and M. Yung, "Scalable group signatures with revocation", In: *Proc. of EUROCRYPT*, CA, USA, pp. 61–76, 2012.

[17] B. Libert, T. Peters, and M. Yung, "Group signatures with almost-for-free revocation", In: *Proc. of CRYPTO*, CA, USA, pp. 571–589, 2012.

[18] B. Wang, L. Baochun, and L. Hui, "Public auditing for shared data with efficient user revocation in the cloud", In: *Proc. of IEEE INFOCOM*, Turin, Italy, pp. 2904–2912, 2013.

[19] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification", In: *Proc. of IEEE INFOCOM*, Toronto, Canada, pp. 2121–2129, 2014.

[20] L. Zhang, Q. Wu, J. D. Ferrer, B. Qin, and Z. Dong, "Round-Efficient and Sender-Unrestricted Dynamic Group Key Agreement Protocol for Secure Group Communications", *IEEE Transactions on Information Forensics and Security*, Vol.10, No.11, pp.2352-2364, 2015.

[21] B. Libert, D.Vergnaud, "Group Signatures with Verifier-Local Revocation and Backward Unlinkability in the Standard Model", *Cryptology and Network Security*, Springer Berlin, Heidelberg, pp.498-517, 2009.

[22] D. Catalano and D. Fiore, "Vector commitments and their applications", *Public-Key Cryptography - PKC 2013*, Nara, Japan, Mar. 2013, pp. 55–72.

[23] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, "Verifiable computation over large database with incremental updates", In: *Proc. of ESORICS*, Wroclaw, Poland, pp. 148–162, 2014.

[24] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, "New publicly verifiable databases with efficient updates", *IEEE Transactions on Dependable and Secure Computing*, Vol.12, No.5, pp.546-556, 2015.

[25] B. Wang, L. Baochun, and L. Hui, "Public auditing for shared data with efficient user revocation in the cloud", In: *Proc. of IEEE INFOCOM*, Turin, Italy, pp. 2904–2912, 2013.

[26] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification", In: *Proc. of IEEE INFOCOM*, Toronto, Canada, pp. 2121–2129, 2014.