ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

Open Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation

Bhuvanavathi Selvi.B[1], Premalatha.J[2]

PG Scholar, Department of CSE, Vandayar Engineering College, Thanjavur, India [1]
Assistant professor, Department of CSE, Vandayar Engineering College, Thanjavur, India [2]

***Abstract*—The approach of the cloud computing makes to store the outsourcing resources turned into a rising pattern, which advances the protected remote information evaluating a debated issue that showed up in the exploration writing. As of late some exploration consider the problem of secure and efficient open information trust worthiness inspecting for shared element information. Be that as it may, these plans are still not secure against the plot of cloud capacity server and renounced gathering client information. To make a productive open trustworthiness inspecting plan with secure gathering client renouncement based on vector responsibility and verifier-nearby renouncement group signature. Outline a solid plan in view of the plan definition. The plan is the general public checking and effective client data sharing furthermore some decent properties, for example, confidently, effectiveness, count ability and traceability of secure group user revocation. At last, the security and exploratory examination demonstrate that, contrasted and its important plans our plan is additionally secure and efficient.**

***Index Terms*— Public integrity auditing, dynamic data, victor commitment, group signature, cloud computing.**

I. Introduction

The continuous development of cloud techniques has boosted a number of public cloud storage applications. Recently, some commercial cloud storage services, such as the simple storage service (S3) [1] on-line data backup services of Amazon and some practical cloud based software Google Drive [2], Drop box [3], Mozy [4], Bitcasa [5], and Memo pal [6], have been built for cloud application. Since the cloud servers may return an invalid result in some cases, such as server hardware/software failure, human maintenance and malicious attack [7], new forms of assurance of data integrity and accessibility are required to protect the security and privacy of cloud user's data.

To overcome the above critical security challenge of today's cloud storage services, simple replication and like Rabin's data dispersion scheme [8] are far from practical application. The formers are not practical because a recent IDC report suggests that data-generation is outpacing storage availability [9]. The later protocols ensure the availability of data when a quorum of repositories, such as k-out-of -n of shared data, is given. However, they do not provide assurances about the availability of each repository, which will limit the assurance that the protocols can provide to relying parties.

For providing the integrity and availability of re-mote cloud store, some solutions [10], [11] and their variants [12], [13], [14], [15], [16], [17], [18] have been proposed. In these solutions, when a scheme supports data modification, we call it *dynamic* scheme, otherwise *static* one (or limited dynamic scheme, if a scheme could only efficiently support some specified operation, such as append). A scheme is *publicly verifiable* means that the data integrity check can be performed not only by data owners, but also by any third-party auditor. However, the dynamic schemes above focus on the cases where there is a data owner and only the data owner could modify the data.

Recently, the development of cloud computing boosted some applications [19], [20], [21], where the cloud service is used as a collaboration platform. In these software development environments, multiple users in a group need to share the source code, and they need to access, modify, compile and run the shared source code at any time and place. The new cooperation network model in cloud makes the re-mote data auditing schemes become infeasible, where only the data owner can update its data. Obviously, trivially extending a scheme with an online data owner to update the data for a group is inappropriate for the data owner. It will cause tremendous communication and computation overhead to data owner, which will result in the single point of data owner. To support multiple user data operation, Wang et al. [22] proposed data integrity based on ring signature. In the scheme, the user revocation problem is not considered and the auditing cost is linear to the group size and data size. To further enhance the previous scheme and support group user revocation, Wang et al. [23] designed a scheme based on proxy re- signatures. However, the scheme assumed that the private and authenticated channels exist between each pair of entities and there is no collusion among them. Also, the auditing cost of the scheme is linear to the group size. Another attempt to improve the previous scheme and make the scheme efficient, scalable and collusion resistant is Yuan and Yu [24], who designed a dynamic public integrity auditing scheme with group user revocation. The authors designed polynomial authentication tags and adopt proxy tag update techniques in

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

their scheme, which make their scheme support public checking and efficient user revocation. However, in their scheme, the authors do not consider the data secrecy of group users. It means that, their scheme could efficiently support plaintext data update and integrity auditing, while not cipher text data. In their scheme, if the data owner trivially shares a group key among the group users, the defection or revocation any group user will force the group users to update their shared key.. In this case, the collusion of revoked user and the cloud server will give chance to malicious cloud server where the cloud server could update the data as many time as designed and provide a legal data finally.

The deficiency of above schemes motivates us to explore how to design an efficient and reliable scheme, while achieving secure group user revocation. To the end, we propose a construction which not only supports group data encryption and decryption during the data modification processing, but also realizes efficient and secure user revocation. Our idea is to apply vector commitment scheme [25] over the database. Then we leverage the Asymmetric Group Key Agreement [26] and group signatures [27] to support cipher text data base update among group users and efficient group user revocation respectively. Specifically, the group user uses the AGKA protocol to encrypt/decrypt the share database, which will guarantee that a user in the group will be able to encrypt/decrypt a message from any other group users.

Our contributions are three folds:

1) We explore on the secure and efficient shared data integrate auditing for multi-user operation for cipher text database.
2) By incorporating the primitives of victor commitment, asymmetric group key agreement and group signature, we propose an efficient data auditing scheme while at the same time providing some new features, such as traceability and count ability.
3) We provide the security and efficiency analysis of our scheme, and the analysis results show that our scheme is secure and efficient.

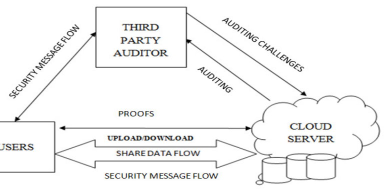

Figure 1. The cloud storage model

A. *Organization*

The rest of this paper is organized as follows: In section 2, we describe the problem formulation. In section 3, we present the used preliminaries. Then, we provide the detail of our scheme in section 4. We conduct the security and efficiency analysis in section 5 and section 6 and leave the related works in section 7. Finally, we show our conclusion in section 8.

II. PROBLEM FORMULATION

In this section, we first describe the cloud storage model of our system. Then, we provide the threat model considered and security goals we want to achieve.

A. *Cloud Storage Model*

In the cloud storage model as shown in Figure 1, there are three entities, namely the cloud storage server, group users and a Third Part Auditor (TPA).

Group users consist of a data owner and a number of users who are authorized to access and modify the data by the data owner. The cloud storage server is semi-trusted, who provides data storage services for the group users. TPA could be any entity in the cloud, which will be able to conduct the data integrity of the shared data stored in the cloud server. In our system, the data owner could encrypt and upload its data to the remote cloud storage server. Also, he/she shares the privilege such as access and modify (compile and execute if necessary) to a number of group users. The TPA could efficiently verify the integrity of the data stored in the cloud storage server; even the data is frequently updated by the group users. The data owner is different from the other group users, he/she could securely revoke a group user when a group user is found malicious or the contract of the user is expired.

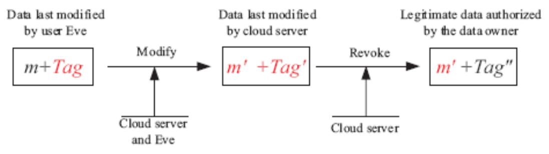
Data last modified by user Eve
Modify
Data last modified by cloud server
Revoke
Legitimate data authorized by the data owner
m+Tag
m' +Tag'
m' +Tag''
Cloud server and Eve
Cloud server

Figure 2. Security problem of server proxy group user revocation

B. *Threat Models and Security Goals*

Our threat model considers two types of attack:

1) An attacker outside the group (include the revoked group user cloud storage server) may obtain some knowledge of the plaintext of the data. Actually, this kind of attacker has to at least break the security of the adopted group data encryption scheme.
2) The cloud storage server colludes with the revoked group users, and they want to provide a illegal data without being detected.

Actually, in cloud environment, we assume that the cloud storage server is semi-trusted. Thus, it is reasonable that a revoked user will collude with the cloud server and share its secret group key to the cloud storage server. In this case, although the server proxy group user revocation way [24] brings much communication and computation cost saving, it will make the scheme insecure against a malicious cloud storage server who can get the secret key of revoked users during the user revocation phase. Thus, a malicious cloud server will be able to make data m, last modified by a user that

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

needed to be revoked, into a malicious data m'. In the user revocation process, the cloud could make the malicious data m' become valid. To overcome the problems above, we aim to achieve the following security goals in our paper:

1) ***Security.*** A scheme is secure if for any database and any probabilistic polynomial time adversary, the adversary cannot convince a verifier to accept an invalid output.

2) ***Correctness.*** A scheme is correct if for any database and for any updated data m by a valid group user, the output of the verification by an honest cloud storage server is always the value m. Here, m is a cipher text if the scheme could efficiently support encrypted database.

3) ***Efficiency***. A scheme is efficient if for any data, the computation and storage overhead invested by any client user must be independent of the size of the shared data.

4) ***Countability.*** A scheme is countable, if for any data the TPA can provide a proof for this misbehavior, when the dishonest cloud storage server has tampered with the database.

5) ***Traceability.*** We require that the data owner is able to trace the last user who updates the data (data item), when the data is generated by the generation algorithm and every signature generated by the user is valid.

III. Preliminaries

Our scheme makes use of bilinear groups. The security of the scheme depends on the Strong Diffie-Hellman assumption and the Decision Linear assumption. In this section, we review the definitions of bilinear groups and the complexity assumption.

A. *Bilinear Groups*

We review a few concepts related to bilinear maps, which follow the notation of [28]. Let G_1 and G_2 be two multiplicative cyclic groups of prime order p, g_1 is a generator of G_1 and g_2 is a generator of G_2. ψ is an efficiently computable isomorphism from G_2 to G_1 with $\psi(g_2) = g_1$, and $e : G_1 \times G_2 \rightarrow G_T$ is a bilinear map with the following properties:

1. **Computability:** there exits and efficiently computable algorithm for computing map e;
2. **Bilinearity:** for all $u \in G_1$, $v \in G_2$ and a, b $\in Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$;
3. **Non-degeneracy:** $e(g_1, g_2)$ **6=** 1.

B. *Complexity Assumption*

The security of our scheme relies on the difficulty of some problems: the Strong Diffie-Hellman problem, the Decision Linear problem, and the Computational Diffie-Hellman problem. We describe these problems as follows.

Definition 1. Q-Strong Diffie-Hellman problem. Let G_1, G_2 be cyclic group of prime order p, where possibly $G_1 = G_2$. Let g_1 be a generator of G_1 and g_2 be a generator of G_2. Given a $(q + 2)$ – tuple$(g1, g2, g_2^{\gamma}, g_2^{(\gamma 2)}, ..., g_2^{(\gamma q)})$ as input, output a pair $(g_1^{1/(\gamma+x)}, x)$ where $x \in Z^*_p$.

The assumption could be used to construct short signature scheme without random oracles [29]. The assumption has properties similar to the Strong-RSA assumption [30] and the properties are adopted for building short group signature in our scheme.

Definition 2. Decision Linear problem. Let g_1 **be** a generator of G_1, and G_1 be a cyclic group of prime order p. Given u, v, h, u^a, u^b,u^c $\in G_1$ as input, output yes if a + b = c and no otherwise.

Definition 3. Square Computational Diffie-Hellman (Square-CDH) problem. With $g \in G_1$ as above, given (g, g^x) for $x \in_R Z_p$ as input, output g^{x2} .

It has been proved that the Square-CDH assumption is equivalent to the classical CDH assumption [32], [33].

Boneh et al. [31] introduced the Decision Linear assumption and they proved that the problem is intractable in generic bilinear groups.

C. *Vector Commitment*

Commitment is a fundamental primitive in cryptography and it plays an important role in security protocols such as voting, identification, zero-knowledge proof, etc. The hiding property of commitment re-quires that it should not reveal information of the committed message, and the binding property re-quires that the committing mechanism should not allow a sender to change his/her mind about the committed message.

Recently, Catalano and Fiore [25] put forward a new primitive called Vector Commitment. Vector Commitment satisfies position binding that an adversary should not be able to open a commitment to two different values at the same position, and the Vector Commitment is concise, which means that the size of the commitment string and its openings have to be independent of the vector length. We provide the formal definition of Vector Commitment [25] as follows.

Definition 4. (Vector Commitment) A vector commitment scheme is a collection of six polynomial-time algorithms (VC.KeyGen, VC.Com, VC.Open, VC.Ver, VC.Update, VC.ProofUpdate) such that:

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.

VC.Com$_{PP}$(m_1, ..., m_q). On input a sequence of q messages m_1, m_q ∈ **M** (**M** is the message space) 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 committee to produce a proof i 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, Λ_i). The verification algorithm accepts (i.e., it outputs 1) only if Λ_i is a valid proof that C was created to a sequence m_1, m_q such that m = m_i.

VC.Update$_{PP}$(C, m, m', i). This algorithm is run by the committee who produces 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 an update information U.

VC.Proof Update$_{PP}$(C, Λ_j , m', i, U). This algorithm can be run by any user who holds a proof Λ_j for some message at position j w.r.t. 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 regard 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.

The primitive of verifiable database with efficient update based on vector commitment is useful to solve the problem of verifiable data outsourcing. Recently, Chen et al. [34], [35] figured out that the basic vector commitment scheme suffers from forward automatic update attack and backward substitution update at-tack. They also proposed a new framework for verifiable database with efficient update from vector commitment, which is not only public verifiable for dynamic outsourced data but also secure against the two attacks. The solution in their scheme is easy to apply in our scheme, which will overcome the attacks they figured out in our scheme.

D. *Group Signature with User Revocation*

We present the formal definition of group signatures with verifier-local revocation [27] as follows.

Definition 5. A verifier-local group signature scheme is a collection of three polynomial-time algorithms (VLR.KeyGen, VLR.Sign, and VLR.Verify), which behaves as follows:

VLR.KeyGen(n). This randomized algorithm takes as input a parameter n, the number of members of the group. It outputs a group public key gpk, an n-element vector of user keys gsk = (gsk[1], gsk[2], ..., gsk[n]), and an n-element vector of user revocation tokens grt, similarly indexed.

VLR.Sign(gpk, gsk[i], M). This randomized algorithm takes as input the group public key gpk, a private key gsk[i], and a message M ∈ {0, 1}*, and returns a signature σ.

VLR.Verify (gpk, RL, σ, M). The verification algorithm takes as input the group public key gpk, a set of revocation tokens RL (whose elements form a subset of the elements of grt), and a purported signature σ on a message M. It returns either valid or invalid. The latter response can mean either that σ is not a valid signature, or that the user who generated it has been revoked.

IV. Scheme Construction

In this section, we provide the formal definition of our scheme according to the definition in [23], [24]. Then, we design the concrete scheme based on our definition.

A. *New Framework*

We consider the database DB as a set of tuple (x, m_x), where x is an index and m_x is the corresponding value. Informally, a public integrity auditing scheme with updates allows a resource-constrained client to outsource the storage of a very large database to a remote server. Later, the client can retrieve and update the database records stored in the server and publicly audit the integrity of the updated data.

B. *Supporting Cipher text Database*

In cloud storage outsourcing environment, the out-sourced data is usually encrypted database, which is usually implicitly assumed in the exiting academic research. Actually, our scheme could support the auditing of database of both plaintext and cipher text database. However, it is not straightforward to extend a scheme to support encrypted database.

In order to achieve the confidentiality of the data record m_x, the client can use his/her secret key to encrypt each m_x using an encryption scheme. When there is only one user (data owner) in the group, the user only needs to choose a random secret key and encrypt the data using a secure symmetric encryption scheme. However, when the scheme needs to support multi-user data modification, while at the same time keeping the shared data encrypted, a shared secret key among group users will result in single point failure problem. It means that any group user (revoked or leave) leak the shared secret key will break the confidentiality guarantee of the data.

To overcome the above problem, we need to adopt a scheme, which could support group user's data modification. Luckily, Wu et al. [26] designed an Asymmetric Group Key Agreement scheme (ASGKA). The scheme has a nice property that, instead of a common secret key, only a shared

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

encryption key is negotiated in an ASGKA protocol. Also, in the scheme, the public key can be simultaneously used to verify signatures and encrypt messages while any signature can be used to decrypt cipher text under this public key. Using the bilinear pairings, the author's instantiate a one-round ASGKA protocol tightly reduced to the decision Bilinear Diffie-Hellman Exponentiation (BDHE) assumption in the standard model. Thus, according to the ASGKA protocol, we consider the case of encrypted database (x, c_x), where x is an index and c_x is the corresponding cipher value.

We provide the detailed changes upon our scheme to support encrypted database.

1) In the **Setup** phase, the scheme has to run the key agreement of ASGKA for the group users.

   Then, the database DB = (i, m_i) is encrypted by the group key gpk of data owner. Finally, the stored database is a cipher text database DB = (i, c_i).

2) In the second step of the **Update** phase, a group user firstly decrypts the record c_i using the AS-GKA secret key gsk[∗] to get plaintext database DB = (i, m_i). Then, update the data to m_i, and later encrypt the data with the public key gpk of ASGKA scheme to get the new encrypted database DB = (i, c_i).

C. *Probabilistic Detection*

Actually, the position binding property of vector commitment of the scheme allows the cloud storage server to prove the data item correctness of certain position. Ateniese et. al. [10] figured out that the sampling ability greatly reduces the overhead on the server and provides high detection probability of server misbehavior. Then, among the q data items, we assume that the third part auditor randomly select x items out of the q-block item database as the target item. In the database, only y items of the database are incorrect. Then, if x, y and q satisfy the specific relationship, the third part auditor could provide a high possession detection ability over the database. The result is interesting that when y is a fraction of the total item number q, the detection probability of server misbehavior is a constant amount of item. For example, if y = 1% of q, then the third part auditor asks for 460 blocks and 300 blocks in order to achieve the detection probability of at least 99% and 95%, respectively.

V. Analysis Of Our Scheme

Our scheme is designed to solve the security and efficiency problems of public data integrity auditing with multi-user modification, where the data has to be encrypted among a dynamic group and any group user can conduct secure and verifiable data update when necessary.

Some basic tools have been used to construct our scheme. Thus we assume that the underlying building blocks are secure, which include the vector commitment, group signature, and asymmetric group key agreement scheme. Based on this assumption, we show that our scheme is secure with respect to the following security analysis

VI. Performance Evaluation

In this section, we provide both the numerical and the experimental analysis of our scheme and conduct the computation time cost comparison with [23] and [24].

A. *Numerical Analysis*

In this section, we conduct the numerical analysis of our scheme and compare the scheme with references [23] and [24].

First of all, all of the three schemes require one-time expensive computational effort in the Setup phase. Then, our scheme is secure against the collusion attack of the cloud storage server and the revoked users in the efficient scheme [24], and it is also efficient, since the computational resources invested by the client is independent on the size of the database. The reason is that, most of the expensive computation overhead is outsourced to the cloud storage server. Finally, the cloud storage server stores the entire database and its relevant materials. Thus, except some private key materials, the group users do not require to store any data locally.

We provide the time cost simulation for our scheme in different phases and the Table 1 presents the numerical analysis of computation of our scheme and two other schemes related. For the convenience of analysis, we denote by Mul a multiplication in G (G_1, G_2 and G_T), Exp an exponentiation in G, Pair a computation of the pairing, and Hash a regular hashing operation. We omit other operations such as addition in G for all the schemes.

Table 1
Performance evaluation and comparison

Scheme	Scheme [23]	Scheme [24]	Our Scheme
Query	-	-	$(q-1)(\mathrm{Mul}+\mathrm{Exp})$
Verify	$2\mathrm{Exp}+\mathrm{Mul}+2\mathrm{Pair}+\mathrm{Hash}$	$\mathrm{Exp}+2\mathrm{Pai}r$	$7\mathrm{Pair}+\mathrm{Mul}+9\mathrm{Exp}+5\mathrm{Hash}$
Update	-	$(s+2)\mathrm{Exp}+(s+1)\mathrm{Mul}$	$s(\mathrm{Mul}+\mathrm{Exp})^*$
ProofUpdate	-	$s\mathrm{Exp}+(s+1)\mathrm{Mul}+c\mathrm{Pair}$	$2s(\mathrm{Mul}+\mathrm{Exp}) or s(\mathrm{Mul}+\mathrm{Exp})^*$
UserRevocation	$(c+d)\mathrm{Exp}+(c+3d)\mathrm{Mul}+(d+1)\mathrm{Pair}+c\mathrm{Hash}$	$c(\mathrm{Pair}+\mathrm{Exp})$	$z(\mathrm{Mul}+2\mathrm{Pair})$

As shown in Table 1, in the **Query** algorithm of our scheme, the computation overhead increases with the database item q. However, we need to remark that the server does not need to compute the proof each time. The reason is that the proof is identical for the same data item and the server only needs to compute once for the first query on each index. Thus, the server could adopt some storage overhead to reduce the computational cost in the **Query** algorithm. Compare with scheme [24], the **Verify** algorithm of our scheme bring much more computation overhead. The reason is that scheme [24] adopts the delegation technology for data updating. In our scheme, to prevent the attack against the collusion of the malicious and revoked group users, we adopt the group signature scheme with secure group user revocation. Although the **Verify** algorithm bring much more computational overhead than scheme [24], it is important that it is a constant part of our scheme. In the **Update** and **Proof**

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

IJARMATE
Your ulti-MATE Research Paper !!!

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

Update algorithms, the computation cost of our scheme and the scheme [24] grow with the increase of data elements (data items in our scheme). For the **User Revocation** algorithm, all the computation time cost grows with the increase of the challenge blocks (items) number. The different is that, scheme [24] is efficient because the computation time cost will not grow with the increase of the group users. Thus, their scheme provides constant computation overhead with different group size. The computation time cost of our scheme grows with the revoked user's number z, which is different from the scheme [23] growing with the increase of the group user's number. Since it is reasonable that the revoked users number z is small than the group users d, we use 2z = d in our simulation.

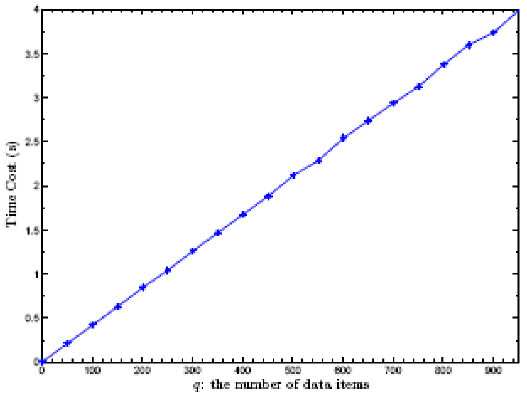

Fig 3 Query Time Cost

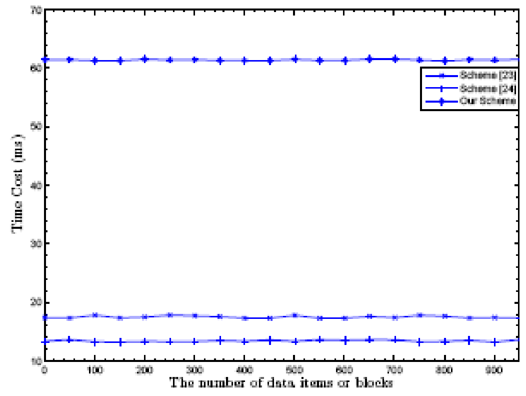

Fig 4 Verify Time Cost

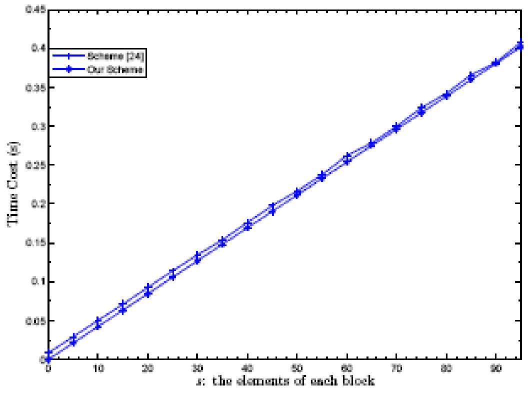

Fig 5 Update Time Cost

B. Experimental evaluation

In this section, we evaluate the thorough experimental evaluation of our scheme. Our experiments are simulated with the pairing-based cryptography library (PBC)[36] on Linux Machine with Intel Core^{TM}2 Duo Processor T9500 running at 2.60GHz and 3G memory. To precisely evaluate the computation complexity at different entities, we simulate all the entity on this machine.

As shown in Figure 3, the **Query** time cost of our scheme is linear with the data items number q, which will take approximately 4 seconds to query about 1000 data items. However, we need to emphasize that the computation cost is at the cloud storage server side, which is very powerful compare with the Linux system running on our laptop. Moreover, the server does not need to run the whole **Query** algorithm every time as analyzed in the previous section.

Actually, in the **Verify** algorithm, the computation overhead mostly comes from the group signature scheme. More precisely, to verify the validity of this phase, we need firstly to verify the integrity of the signature, which means that our scheme need to generate the time costed parameters such as R_1, R_2, and R3. Actually, the computation time cost of our scheme a constant number. Also, it is around 5 times that of the most efficient scheme [24]. In the Figure 5, we show the data update computation comparison with scheme [24], and both their computation overheads grow with the increase of the element number in each block.

In the **Proof Update** algorithm, as shown in Figure 6, the computation time cost grows with the increase of the elements number of each block and the number of selected challenging data blocks. Actually, the data blocks contain data elements in scheme [24] while not in our scheme. It is interesting that, if we do not consider the data elements in the scheme [24], the computation overhead of the two schemes in with the same challenging number are almost the same.

The **User Revocation** algorithm simulation in Figure 7 shows that scheme [24] is the most efficient one. Compare with scheme [23] whose computation overhead grows rapidly with the increase of group users number and the selected challenging blocks number, scheme [24] and our scheme have

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

a flat-ting growth. The reason is that, scheme [23] has to consider the whole group users number, while the computation time cost in our scheme is related to the revoked group users. It means that we need to verify e $(T_2/A, u\hat{})$ = e $(T_1, v\hat{})$ for each user in the revocation list. The best scheme is [24], whose computation overhead is irrelevant to the group user's number. They achieve this by allowing the cloud storage server to recompute the authentication tag of blocks last modified by a revoked group user. We analyze this tag update delegation way in the previous section and point out that it is not secure against the cloud storage server and revoked group users collusion attack.

VII. Related Work

Plenty of researchers have devoted considerable attention to the problems on how to securely outsource local store to remote cloud server. Among which, the problem of remote data integrity and availability auditing attacks the attestation of many researchers. The concepts and solution Provable Data Possession (PDP) and Proofs of Retrievability (PoR) were first proposed by Ateniese et al. [10] and Juels et al. [11]. In their scheme, the homomorphic authentication technique was adopted to reduce both the communication and computation cost. Later, a number of variants of PDP and PoR schemes are designed to improve the efficiency and enhance the function of basic schemes, such as allowing public auditing [16], [22], [17] and supporting data update [14], [15].

To enhance the previous works, Wang et al. [22] designed a scheme to support share data integrity auditing, whose scheme adopted ring signature to protect the privacy of users. The limitation of the scheme is that it does no support dynamic group and also suffers from a computational overhead linear to the group size and the number of data auditing. To further support user revocation, Wang et al. [23] designed another scheme based on the assumption that no collusion occurs between cloud servers and revoked user. As a matter of fact, they assumed that the private and authenticated channels exit between each pair of entities and collusion between invalid users and cloud servers will lead to the disclosure of secrets of all other valid users. Recently, Yuan and Yu [24] designed a dynamic public integrity auditing scheme with secure group user revocation. The scheme is based on polynomial authentication tags and adopts proxy tag update techniques, which makes their scheme support public checking and efficient user revocation. However, the authors do not consider the cipher text store. Also, to make the scheme efficient, the data owner (the data owner's private key is not necessary) does not take part in the user revocation phase, where the cloud could conduct some malicious operation of user's data when it colludes with the revoked users.

Gennaro et al. [37] formalized the notion of verifiable computation which allows a client to outsource the computation of an arbitrary function. However, it is inefficient for practical applications due to the complicated fully homomorphic encryption techniques [38], [39]. Also, another disadvantage of the schemes based on fully homomorphic encryption is that, the client must repeat the expensive pre-processing stage if the malicious server tries to cheat and learn a bit of information. Benabbas et al. [40] proposed the first practical verifiable database scheme based on the hardness of the subgroup membership problem in bilinear groups with Composite order. However, the scheme does not support the public verifiability property. Catalano and Fiore [25] proposed a practical solution to build verifiable database (VDB) from vector commitment that supports the public verifiability. Both of the schemes assume that the size of the out-sourced database should be fixed and the client can know the outsourcing function in advance. Recently, Backes et al. [41] presented a flexible VDB scheme with two additional properties that eliminates the assumption.

Group signature is introduced by Chaum and Heyst [42]. It provides anonymity for signers, where each group member has a private key that enables the user to sign messages. However, the resulting signature keeps the identity of the signer secret. Usually, there is a third party that can conduct the signature anonymity using a special trapdoor. Some systems support revocation [43], [44], [45], [27], [46], [47], where group membership can be disabled with-out affecting the signing ability of unrevoked users. Boneh and Shacham [27] proposed an efficient group signature with verifier-local revocation. The scheme provides the properties of group signature such as selfless-anonymity and traceability. Also, the scheme is a short signature scheme where user revocation only requires sending revocation information to sig-nature verifiers. Libert et al. [46] proposed a new scalable revocation method for group signature based on the broadcast encryption framework. However, the scheme introduces important storage overhead at group user side. Later, Libert et al. [47] designed a scheme to enhance the former scheme which could obtain private key of constant size. In their scheme, the unrevoked members still do not need to update their keys at each revocation.

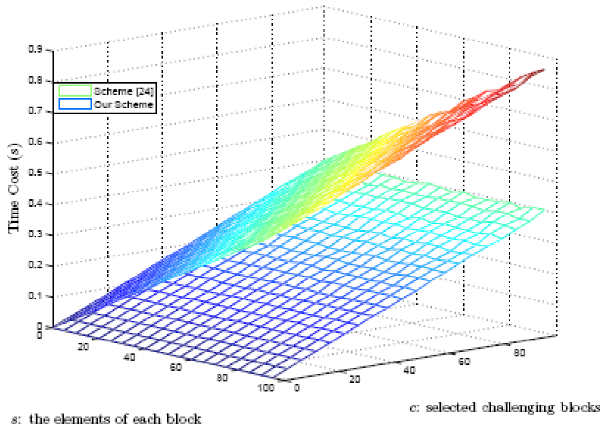

Fig 6 Proof Update Time Cost

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com

International Journal of Advanced Research in Management, Architecture, Technology and Engineering (IJARMATE)
Vol. 2, Issue 4, April 2016

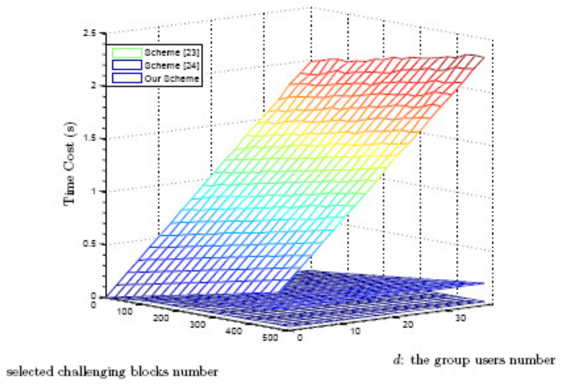

Fig 7 User Revocation Time Cost

VIII. Conclusion

The primitive of verifiable database with efficient updates is an important way to solve the problem of verifiable outsourcing of storage. We propose a scheme to realize efficient and secure data integrity auditing for share dynamic data with multi-user modification. The scheme vector commitment, Asymmetric Group Key Agreement (AGKA) and group sig-natures with user revocation are adopt to achieve the data integrity auditing of remote data. Beside the public data auditing, the combining of the three primitive enable our scheme to outsource cipher text database to remote cloud and support secure group users revocation to shared dynamic data. We provide security analysis of our scheme, and it shows that our scheme provide data confidentiality for group users, and it is also secure against the collusion attack from the cloud storage server and revoked group users. Also, the performance analysis shows that, compared with its relevant schemes, our scheme is also efficient in different phases.

References

[1] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the weil pairing,” in *Proc. of Asiacrypt 2001*, Gold Coast, Australia, Dec. 2001, pp. 514–532.
[2] D. Boneh and X. Boyen, “Collision-free accumulators and fail-stop signature schemes without trees,” in *Proc. of EUROCRYPT 2004*, Interlaken, Switzerland, May 2004, pp. 56–73.
[3] D. Boneh, X. Boyen, and H. Shacham, “Short group signatures,” in *Proc. of CRYPTO 2004*, CA, USA, Aug. 2004, pp. 41–55.
[4] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, “Verifiable computation over large database with incremental updates,” in *Proc. of ESORICS 2014*, Wroclaw, Poland, Sep. 2014, pp. 148–162.
[5] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, “New publicly verifiable databases with efficient updates,” to appear in IEEE Transactions on Dependable and Secure Computing, Accepted.
[6] R. Gennaro, C. Gentry, and B. Parno, “Non-interactive verifiable computing: Outsourcing computation to untrusted workers,” in *Proc. of CRYPTO 2010*, CA, USA, Sep. 2010, pp. 465– 482.
[7] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in *Proc. of ACM STOC 2009*, Washington DC, USA, May 2009, pp. 169–178.
[8] A. Juels and B. S. Kaliski, “Pors: Proofs of retrievability for large files,” in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 584–597.
[9] K. D. Bowers, A. Juels, and A. Oprea, “Proofs of retrievability: theory and implementation,” in *Proc. of CCSW 2009*, llinois, USA, Nov. 2009, pp. 43–54.
[10] Y. Dodis, S. Vadhan, and D. Wichs, “Proofs of retrievability via hardness amplification,” in *Proc. of TCC 2009*, CA, USA, Mar. 2009, pp. 109–127.
[11] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Proofs of retrievability via hardness amplification,” in *Proc. of ESORICS 2009*, Saint-Malo, France, Sep. 2009, pp. 355–370.
[12] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, “Dynamic provable data possession,” in *Proc. of ACM CCS*, Illinois, USA, Nov. 2009, pp. 213–222.
[13] 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 2010*, CA, USA, Mar. 2010, pp. 525– 533.
[14] E. Shi, E. Stefanov, and C. Papamanthou, “Practical dynamic proofs of retrievability,” in *Proc. of ACM CCS 2013*, Berlin, Germany, Nov. 2013, pp. 325–336.
[15] C. Gentry and S. Halevi, “Implementing gentrys fully-homomorphic encryption scheme,” in *Proc. of EUROCRYPT 2011*, Tallinn, Estonia, May 2011, pp. 129–148.
[16] S. Benabbas, R. Gennaro, and Y. Vahlis, “Verifiable delegation of computation over large datasets,” in *Proc. of CRYPTO 2011*, CA, USA, Aug. 2011, pp. 111–131.
[17] M. Backes, D. Fiore, and R. M. Reischuk, “Verifiable delegation of computation on outsourced data,” in *Proc. of ACM CCS 2013*, Berlin, Germany, Nov. 2013, pp. 863–874.
[18] D. Chaum and E. van Heyst, “Group signatures,” in *Proc. of EUROCRYPT 1991*, Brighton, UK, Apr. 1991, pp. 257–265.
[19] E. Bresson and J. Stern, “Efficient revocation in group signatures,” in *Public-Key Cryptography - PKC 2001*, Cheju Island, Korea, Feb. 2001, pp. 190–206.
[20] J. Camenisch and A. Lysyanskaya, “Dynamic accumulators and application to efficient revocation of anonymous credentials,” in *Proc. of CRYPTO 2002*, CA, USA, Aug. 2002, pp. 61–76.
[21] G. Ateniese, D. Song, and G. Tsudik, “Quasi-efficient revocation in group signatures,” in *Proc. of FC 2002*, Soughamption, Bermuda, Mar. 2002, pp. 183–197.
[22] B. Libert, T. Peters, and M. Yung, “Scalable group signatures with revocation,” in *Proc. of EUROCRYPT 2012*, CA, USA, Aug. 2002, pp. 61–76.