I J C T A, 9(16), 2016, pp. 7703-7709
© International Science Press

Integrity Auditing with Homomorphic Encryption for Dynamic Cloud with Group user Revocation

Madhuri P.* and Jayavarthini C.**

ABSTRACT

With cloud computing, data owners are motivated to outsource their data from local sites to public cloud for great flexibility and economic saving. Recently, some research consider the problem of secure and efficient public data integrity auditing for shared dynamic data. But this scheme is not secure against collusion of cloud storage server. An efficient public integrity auditing with a secured group user revocation based on vector commitment and group user revocation. A distributed key generation algorithm is used to generate authenticated userpasswords across multiple servers and eliminate single point failures. This scheme supports the public checking and efficient user revocation and also provides confidentiality, efficiency andtraceability of secure group user revocation. A homomorphic encryption algorithm is also used forcreating unique id for the users.

Keywords: data integrity, vector commitment, homomorphic encryption, integrity auditing.

1. INTRODUCTION

Cloud computing is an on-demand computing, and is also a kind of Internet-based computing which provides shared processing resources and also the data to computers and other devices on demand. This is a model for enabling on-demand access to the shared pool of a configurable computing resources for e.g., networks, servers, storage, applications and services, which can be rapidly released with a minimal management effort. Cloud computing and the storage solutions provides the users as well as enterprises with a various capabilities to store and process their data in the third-party data centres. It depends on the sharing of resources in order to achieve coherence and economy of scale which is similar to a utility over a network.

Considering the authenticity of data, it is emerged as a critical issue in storing data on untrusted servers. It occurs in the peer-to-peer storage systems, long-term archives, database systems. Those systems prevent servers of storage from misrepresenting and modifying data which provides authenticity that checks while accessing data. A model for provable data possession (PDP) allows the client who stored data in an untrusted server to verify that the server possesses the original data without retrieving it [1]. A POR(Proofs Of Retrievability for large files) enables a back-up service and archive and to produce a proof that a user can retrieve a target file, that is, that the archive retains and reliably transmits file data sufficient for the user to recover the file in its entirety A POR may be viewed as a kind of cryptographic proof of knowledge, but it is the one, which is specially designed to handle a large file F or bit string[2].

As PORs incur lower communication complexity than transmission of F itself, they are an attractive building block for high-assurance remote storage systems [3]. Most existing POR schemes however, are impractical due to the prohibitive costs associated with data updates or client storage. A recent POR construction by Cash et al shows how to achieve asymptotic efficiency in the presence of updates with

* Department of Computer Science & Engineering, SRM University, Chennai.

** Assistant Professor in Department of Computer Science & Engineering, SRM University, Chennai.

constant client storage— however, their scheme relies on Oblivious RAM (ORAM), a rather heavy-weight cryptographic primitive [4]

As storage-outsourcing services and resource-sharing networks have become popular, the problem of efficiently proving the integrity of data stored at untrusted servers has received increased attention. . In the provable data possession (PDP) model, the client pre process the data and then sends it to an untrusted server for storage, while keeping a small amount of meta-data, the client later asks A group key agreement is a protocol, that allows a set of users to establish a common secret information via open networks. By observing that, a major goal of Group Key

Agreement is to establish a confidential channel among group members. This encryption key may be accessible to the attackers and it corresponds to different decryption keys, in each of which is only computable by one group member [7].

By enabling the public auditing for cloud data storage, the security is of critical importance so that the users may resort to an external auditing party, for checking the integrity of outsourced data whenever needed. To introduce an effective third party auditor, following are the two fundamental requirements that have to be met: 1. Third party auditor should be able to efficiently audit the cloud data storage without acquiring the local copy of data, and it won't introduce any additional on-line burden to the cloud user. 2. The third party auditing process should bring no new vulnerabilities towards user data privacy [8].

2. PROPOSED SYSTEM

2.1. Problem

The third party auditor can view all the data which is exchanged between cloud users as well as with the cloud server, which is not necessary. TPA will maintain the history of the data users in the cloud and also the actions performed by them.

2.2. Objective

An advanced scheme to support stronger security by encrypting the file with different privilege keys is presented, and also to enhance the security of deduplication and protect the data confidentiality. The TPA should only deal with the privilege issues and it is not necessary for him to view all the data exchange between cloud users of that respective group.

2.3. Solution

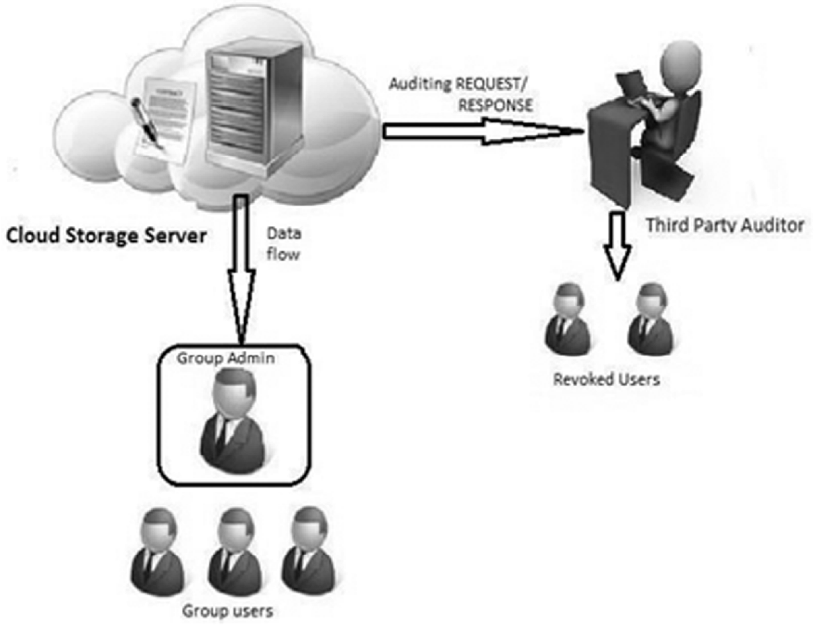
Advanced Encryption Standard (AES) is the best algorithm for secure data storage in this function performs the searching and sorting of the similar data items in the cloud domain. AES algorithm uses a key size of 128 bits, 192 bits or 256 bits. Vector Commitment, a six polynomial algorithm is used to update the information in the cloud. Group Signature algorithm is a 3 polynomial time algorithms, which are used to generate a key, signature and verification. A distributed key generation (DKG) is an encryption process in which multiple parties contribute to the calculation of a shared public and private key set. A Homomorphic Encryption technique is also used in order to increase the integrity of the shared data.

3. ARCHITECTURAL DIAGRAM

4. ALGORITHMS

4.1. AES (Advanced Encryption Standards)

It is an iterative rather than Feistel cipher. This comprises of a series of linked operations, which involve replacing inputs by specific outputs or substitutions and others involve shuffling bits around so called permutations.

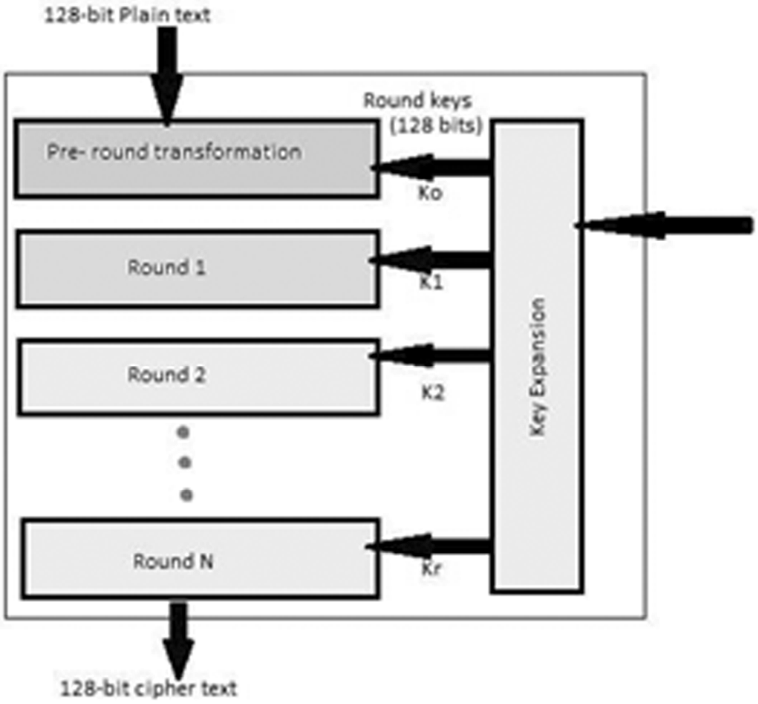


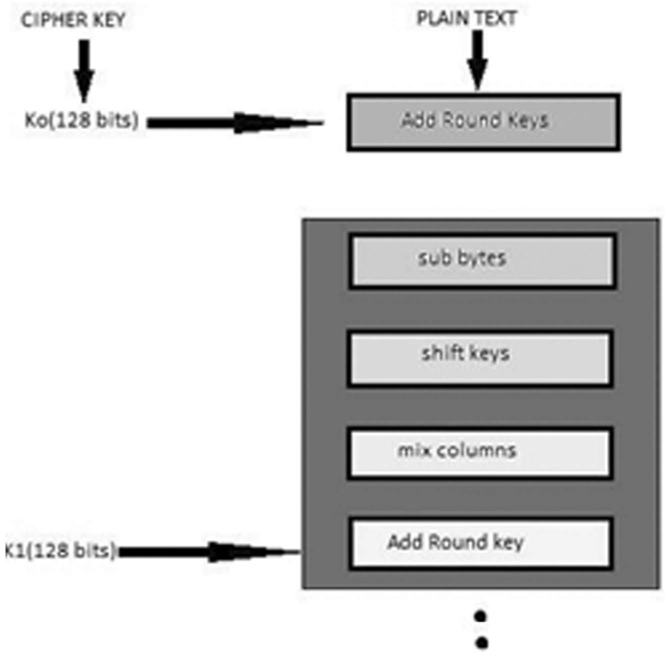
AES performs all its computations on bytes rather than bits. Hence, AES takes a 128 bit secret key and it will be combined with a plaintext block which is arranged in four columns and four rows for processing as a matrix. This is called cipher text. But in DES, the number of rounds is variable and they depends on the length of the key. AES uses 10 rounds for 128-bit keys, and it takes 9 loops for 10 rounds. Likewise 12 rounds for 192-bit keys with 11 loops for 12 rounds.

And finally 14 rounds for 256-bit keys, with 13 loops for 14 rounds. Every round uses a different 128-bit,192-bit and 256-bit round key respectively, which is calculated from the original AES key

4.1.1. Encryption Process

The description of a typical round of AES encryption is given below. Each round comprise of four sub-processes. The first round process is depicted below:





1) Byte Substitution: (Sub Bytes) 16 input 2) Shift rows: Each four rows of the matrix are shifted to the left. Shift is carried out as below-

- First row is not shifted.
- Second row is shifted to one (byte) position to the left.
- And third row is shifted two positions to the left from right.
- Fourth row is shifted three positions to the left.
- The resulting is a new matrix consisting of the 16 bytes, but shifted with respect to each other.

2) Mix Columns: Each column of four bytes is converted by using a special mathematical function. This function takes an input of four bytes of one column and produces an output, which is a four completely new bytes, which replaces the original column. And the result will be an another new matrix, which consists of 16 new bytes. It should be noted that this step is not performed in the last round.

3) Add round key: The 16 bytes of the matrix are now considered as 128 bits and are XOR to the 128 bit round key. If this is the last round then the output is the cipher text. Or else the resulting 128 bits are interpreted as 16 bytes and another similar round.

4.1.2. Decryption Process

The decryption process for an AES cipher text is exactly the reverse order of the encryption process. Each round consists of the four processes taken place in the reverse order:

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since the decryption process is in reverse manner, the encryption and decryption algorithms should be separately implemented, although they are related very closely.

4.1.3. AES Analysis

In the present day cryptography, AES is adopted widely and is also supported in both the hardware and software. Till now, no practical cryptanalytic attacks are there against AES.

AES has a built in flexibility of the key length, so it allows a degree of future-proofing which is against progress in the ability to perform exhaustive key searches.

Advanced Encryption Standard (AES) is six time faster than DES. So a replacement for DES was needed, as its key size is too small. As the computing power is increasing, it became vulnerable against exhaustive key search attack. So Triple DES is designed to overcome that drawback but it was found very slow.

The features of AES are as follows:

- A Symmetric key and a symmetric block cipher.
- It has a 128-192-256 bit keys.
- Stronger and faster than the Triple DES.
- Provide full specification and design details

4.2. Vector commitment

The hiding property of commitment requires that should not reveal the information of committed message, and the binding property requires that the committing mechanism should not allow a sender to change their mind regarding the committed message.

A vector commitment scheme is a collection of six polynomial-time algorithms (VC. Key Generation, VC. Commit, VC. Open, VC. Verify, VC. Update, VC. Proof Update) such that:

1) VC. Key Generation (1k, q): It is Given the security parameter k and the size q of the committed vector the key generation outputs some public parameters pp.
2) VC. Compp (m1, ..., mq): The input is a sequence of q messages m1, m2, m3 ..., m M where M is the message space and the public parameters pp, the committing algorithm produces an output commitment string C and also an auxiliary information aux.
3) VC. Openpp (m, i, aux): The algorithm is run by the committer to create a proof i that m is the i-th committed message. In this case, notice that when some of the updates have occurred,then the auxiliary information aux can include the update information produced by these updates.
4) VC. Verpp (C, m, i, Δi): The verification algorithm accepts (i.e., it outputs 1) only if Δi is a valid proof that C was created to a sequence m1, m2, m3 ....., mq so that m = mi.
5) VC. Updatepp (C, m, m2, i): This algorithm runs by the committer who produces C and also who wants to update it by converting the i-th message to m2. The algorithm takes an input of the old message m, and then it produces an output as a new message m2 in the position i. It outputs a new commitment called C along with the updated information U.
6) VC. Proof Update (m, I, U): The algorithm takes as input the old message m, the new message m2 and the position i. It outputs a new commitment C together with an update information U.

4.3. Group Signature with User Revocation

A group signature scheme consists of three polynomial-time algorithms which are explained below:

1) VLR. Key Generation (n). It is a randomized algorithm, which takes an input parameter n, and the number of members of the group. This produces an output of a group public key gpk, an n-element

vector of user keys gsk = (gsk[1], gsk[2], gsk[3]. gsk[4] ..., gsk[n]), and also an n-element vector for user revocation tokens grt, which are similarly indexed.

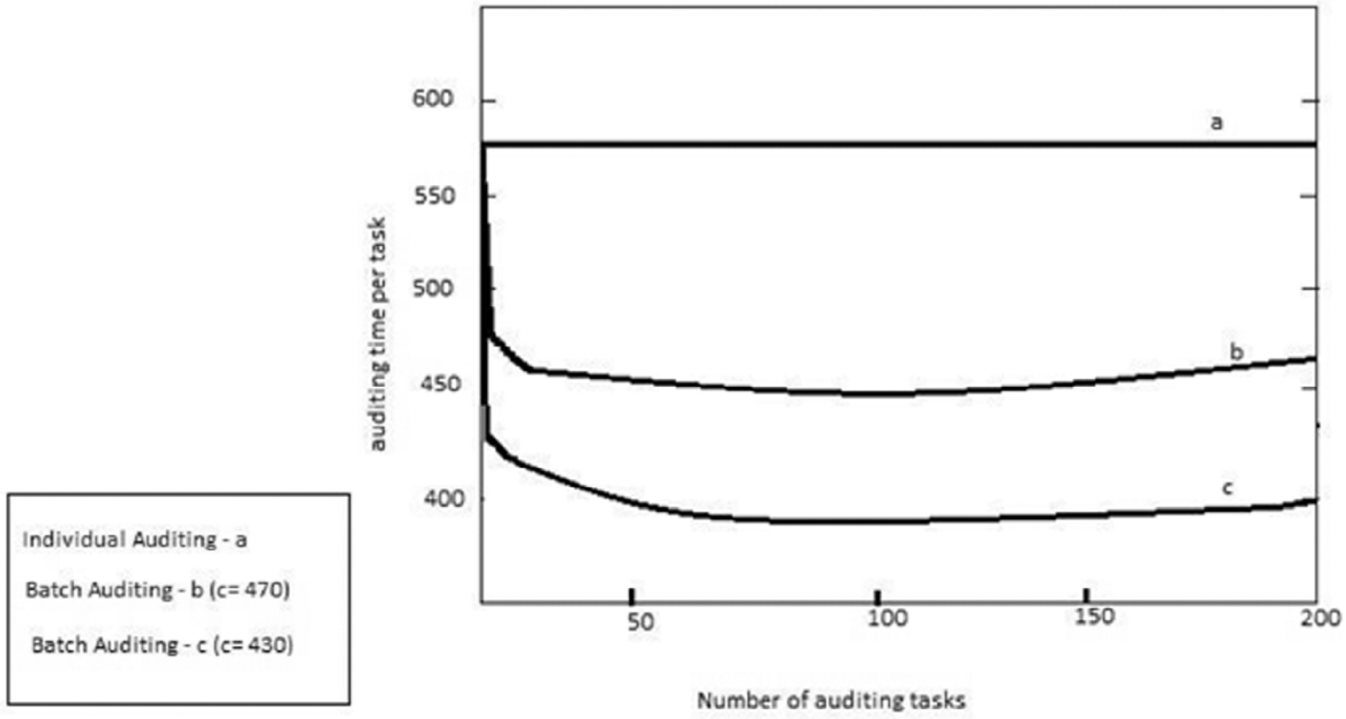
2) VLR.Sign (gpk, gsk [i], M). This randomized algorithm takes as input the group public key gpk, and also a private key called gsk[i], with a message M $\in$ {0, 1}
3) VLR.Verify (gpk, RL, σ, M). The verification algorithm takes an input as the group public key gpk,with a set of revocation tokens RL and a purported signature ó on a message M. The parameters which are returned are either valid or invalid. The latter response can mean either that sigma -σ and it is not a valid signature, or that the user who generated it has been revoked.

5. RESULTS & DISCUSSION

Plenty of researchers have devoted considerable attention to the problems on how to securely outsource local store to remote cloud server. The problem of remote data integrity and availability auditing attacks the attestation of many researchers, to enhance the previous works, Wang et al designed a scheme in order to support and share the data integrity auditing, and this scheme adopts a ring signature to protect the privacy of users.

The limitation of this scheme is that, it does not support any dynamic group and also suffers from a high computational overhead which is linear to the group size and the number of data auditing. So in order to further support user revocation, Wang et al designed a scheme based on the assumption that no collusion occurs between cloud servers and revoked user.

Group signature provides anonymity for signers, where each group member has a private key which will enable the user to sign in the messages. Since, the resulting signature keeps the identity of the signer secret. Usually, there is a third party that can conduct the signature anonymity. Few systems support revocation, where group membership can be disabled without affecting the signing ability of unrevoked users.



Comparison of auditing time between batch auditing and individual auditing.

REFERENCES

[1] “Above the Clouds: A Berkeley View of Cloud Computing”, Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee.

[2] “Provable Data Possession at Untrusted Stores”, Giuseppe Ateniese, Randal Burns Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, Dawn Song.

[3] “PORs: Proofs of Retrievability for Large Files”, AriJuels and Burton S. Kaliski Jr.

[4] “Practical Dynamic Proofs of Retrievability”, Elaine Shi, Emil, Stefanov.

[5] “Dynamic Provable Data Possession”, C. Chris ErwayAlptekinKupc, Charalampos Papamanthou, Roberto Tamassia

[6] “Efficient Public Integrity Checking for Cloud Data Sharing with Multi-User Modification”, Jiawei Yuan.

[7] “Asymmetric Group Key Agreement”, Qianhong Wu1, 2, Yi Mu3, Willy Susilo3, Bo Qin1, 4 and Josep Domingo-Ferrer1

[8] “Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing”, Cong Wang, Qian Wang, and KuiRen, Wenjing Lou.