IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 4 Issue 5, May 2017
ISSN (Online) 2348 – 7968 | Impact Factor (2016) – 5.264
www.ijiset.com

DeyPos: Using Revocable Storage for Multi-User Environments Identity-Based Encryption

P.RAJU, M.Phil Scholar, M.Phil Scholar,.
A.SENTHIL KUMAR, Assistant Professor,
Tamil University, Thanjavur, Tamil Nadu, India.

Abstract

Dynamic Proof of Storage (PoS) is a useful cryptographic primal that allow a user to ensure the reality of Outsourced files and to resourcefully modernize the files in a cloud server. A realistic multi-user cloud storage system wants the safe client-side cross-user deduplication technique, which allows a user to skip the uploading process and obtain the ownership of the files immediately, when other owners of the same files have uploaded them to the cloud server. Cloud computing present a flexible and fitting way for data sharing, which brings various benefits for both the society and individuals. But there exists a natural resistance for users to directly outsource the shared data to the cloud server since the data often contain valuable information. It is necessary to place cryptographically enhanced access control on the shared data. Identity-based encryption is a promising cryptographically primitive to build a practical data sharing system. Revocable-storage identity-based encryption (RSIBE), which can provide the forward/backward safety of cipher-text by introducing the functionalities of user revocation and cipher-text update concurrently. Furthermore, we present a concrete construction of RS-IBE, and prove its security in the defined security model. The performance association points to that the planned RS-IBE scheme has advantages in terms of **functionality** and competence, and thus is feasible for a practical and cost-effective data-sharing system.

INTRODUCTION

Data integrity is one of the most important properties when a user outsources its files to cloud storage. Users should be convinced that the files stored in the server are not tampered. raditional techniques for protecting data integrity, such as message authentication codes (MACs) and digital signatures, require users to download all the files from the cloud server for verification, which incurs a heavy communication cost. These techniques are not suitable for cloud storage services where users may check the integrity frequently, such as every hour. Thus, researchers introduced *Proof of Storage* (PoS) for checking the integrity without downloading files from the cloud server. Furthermore, users may also require several dynamic operations, such as modification, insertion, and deletion, to update their files, while maintaining the capability of PoS. Dynamic PoS is proposed for such dynamic operations. In contrast with PoS, dynamic PoS employs authenticated structures, such as the Merkle tree. Thus, when dynamic operations are executed, users regenerate tags (which are used for integrity checking, such as MACs and signatures) for the updated blocks only, instead of regenerating for all blocks. To better understand the following contents, they present more details about PoS and dynamic PoS.While outsourcing data to cloud server, users also want to control access to these data such that only those currently authorized users can share the

outsourced data. A natural solution to conquer the afore mentioned problem is to use cryptographically enforced access control such as identity-based encryption (IBE). Furthermore, to overcome the above security threats, such kind of identitybased Right to use organizes placed on the joint data should meet the following protection goals:

A. Data Confidentiality
Unauthorized users should be prevented from accessing the plaintext of the shared data stored in the cloud server. In addition, the cloud server, which is supposed to be honest but curious, should also be deterred from knowing plaintext of the shared data.

B. Backward Secrecy
Backward secrecy means that, when a user's authorization is expired, or a user's secret key is compromised, he/she should be prohibited from accessing the plaintext of the consequently shared data that are motionless encrypted under his/her characteristics.

C. Forward Secrecy
Forward secrecy means that, when a user's authority is expired, or a user's secret key is compromised, he/she should be prevented from accessing the plaintext of the shared data that can be previously accessed by him/her.

3. REVIEW OF LITERATURE

The objective of this literature review is to study the work carried and published by different researchers and authors in the domain of Document Annotation and tagging.

Kun He, Jing Chen, Ruiying Du, Qianhong Wu, Guoliang Xue, and Xiang Zhang proposed in "DeyPoS: Deduplicatable

Dynamic Proof of Storage for Multi-User Environments" the comprehensive requirements in multi-user cloud storage systems and introduced the model of deduplicatable dynamic PoS. They designed a novel tool called **HAT which is an efficient authenticated structure. Based on HAT, proposed the first practical deduplicatable dynamic PoS scheme called DeyPoS** and confirmed its protection in the arbitrary oracle model. The imaginary and investigational results prove that our DeyPoS implementation is efficient, especially when the file size and the number of the challenged blocks are large.[1]

Jianghong Wei, Wenfen Liu, Xuexian Hu proposed in paper "Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption" confirm cloud computing brings huge handiness for people. Particularly, it perfectly matches the increased need of sharing data over the Internet. In this paper, to construct a commercial and protected data distribution system in cloudcomputing, they proposed a notion called RS-IBE, which supports identity revocation and ciphertext update simultaneously such that a revoked user is prevented from accessing previously shared data, as well as subsequently shared data. Furthermore, a concrete construction of RS-IBE is presented. [2]

Pietro and Sorniotti proposed in paper "Boosting Efficiency and Security in Proof of Ownership for Deduplication" proves another proof of ownership scheme which improves the efficiency. Xu et al.[4] proposed a client-side deduplication scheme for encrypted data, but the scheme employs a deterministic proof algorithm which

indicates that every file has a deterministic short proof. Thus, anyone who obtains this proof can pass the verification without possessing the file locally. Other deduplication schemes for encrypted data were proposed for enhancing the security and efficiency. Note that, all existing techniques for cross-user deduplication on the client-side were designed for static files. Once the files are updated, the cloud server must regenerate thecomplete authenticated structures for these files, which causes heavy computation cost on the server-side. [3]

The concept of proof of storage was introduced by Ateniese et al. in paper "Provable data possession at untrusted stores", and Juels and Kaliski, respectively. The main idea of PoS is to randomly choose a few data blocks as the challenge. Then, the cloud server returns the challenged data blocks and their tags as the response. Since the data blocks and the tags can be combined via homomorphic functions, the communication costs are reduced. The subsequent works extended the research of PoS, but those works did not take dynamic operations into account. Erway et al. and later works focused on the dynamic data. Among them, the scheme in is the most efficient solution in practice. However, the scheme is stateful, which requires users to maintain some state informationof their own files locally. Hence, it is not appropriate for a multiuser environment. Halevi et al. introduced the concept of proof of ownership which is a solution of cross-user deduplication on the client-side. It requires that the user can generate the Merkle tree without the help from the cloud server, which is a big challenge in dynamic PoS. [5]

Zheng and Xu proposed in paper "Secure and efficient proof of storage with deduplication" proves a explanation called proof of storage with deduplication, which is the first challenge to intend a PoS method with deduplication. Du et al. Introduced proofs of ownership and retrievability, which are like but more efficient in terms of computation cost. Note that neither can support dynamic operations.Due to the problem of structure diversity and private tag generation, cannot be extended to dynamic PoS. Wang et al. and Yuan and Yu considered proof of storage for multi-user updates, but those schemes focus on the problem of sharing files in a group. Deduplication in these scenarios is to deduplicate files among different groups. Unfortunately, these schemes cannot support deduplication due to configuration diversity and private ticket production. In this paper, they consider a more general situation that every user has its own files separately. [6]

Jingwei Li, Jin Li, Dongqing Xie, and Zhang Cai "Secure Auditing and Deduplicating Data in cloud" proves both data integrity and deduplication in cloud, they propose SecCloud and SecCloud+. SecCloud introduces an auditing entity with maintenance of a MapReduce cloud, which assist clients create data tags earlier than uploading as well as check the integrity of data having been stored in cloud. In addition, SecCloud enables secure deduplication through introducing a PoS protocol and preventing the leakage of side channel information in data deduplication. Compared with previous work, the computation by user in SecCloud is greatly reduced during the file uploading and auditing phases. SecCloud+ is an advanced construction motivated by the fact that customers always would like to encrypt their data prior to uploading, and agree to for integrity auditing and protected deduplication directly on encrypted data.

Compared with previous work, the calculation by user in SecCloud is greatly reduced during the file uploading and auditing phases[7].

4. DEDUPLICATABLE DYNAMIC POS

No trivial extension of dynamic Pos can achieve cross-user deduplication. To fill this void, we present a novel primitive called deduplicatable dynamic Proof of storage in this section.

System Model

Our system model considers two types of entities: the cloud server and users, as shown in Fig. 2. For each file, *original user* is the user who uploaded the file to the cloud server, while *subsequent user* is the user who proved the ownership of the file but did not actually upload the file to the cloud server. There are five phases in a deduplicatable dynamic PoS system: *pre-process*, *upload*, *deduplication*, *update*, and *proof of storage*.

In the *pre-process* phase, users intend to upload their local files. The cloud server decides whether these files should be uploaded. If the upload process is granted, go into the upload phase; otherwise, go into the deduplication phase.

In the *upload* phase, the files to be uploaded do not exist in the cloud server. The original users encodes the local files and upload them to the cloud server.

In the *deduplication* phase, the files to be uploaded already exist in the cloud server. The subsequent users possessthe files locally and the cloud server stores the authenticated structures of the files. Subsequent users need to convince the cloud server that they own the files without uploading them to the cloud server.Note that, these three phases (pre-process, upload, and deduplication) are executed only once in the life cycle of a file from the perspective of users. That is, these threephases appear only when users intend to upload files. If these phases terminate normally, i.e., users finish uploading in the upload phase, or they pass the verification in the deduplication phase, we say that the users have the ownerships of the files.

In the *update* phase, users may modify, insert, or delete some blocks of the files. Then, they update the corresponding parts of the encoded files and the authenticated structures in the cloud server, even the original files were not uploaded by themselves. Note that, users can update the files only if they have the ownerships of the files, which means that the users should upload the files in the upload phase or pass the verification in the deduplication.phase. For each update, the cloud server has to reserve the original file and the authenticated structure if there exist other owners, and record the updated part of the file and the authenticated structure. This enables users to update a file concurrently in our model, since each update is only “attached” to the original file and authenticated structure.

In the *proof of storage* phase, users only possess a small constant size metadata locally and they want to check whether the files are faithfully stored in the cloud server Without downloading them. The files may not be uploaded by these users, but they pass the deduplication phase and prove that they have the ownerships of the files. Note that, the update phase and the proof of storage Phase can be executed multiple times in the life cycle of a file. Once the ownership is

verified, the users can arbitrarily enter the update phase and the proof of storage phase Without keeping the original files locally.

5 . THE CONSTRUCTION OF DEYPOS

We propose a concrete scheme of deduplicatable dynamic PoS called DeyPoS. It consists of five algorithms as described in Section 2: Init, Encode, Deduplicate, update, and Check.

4.1 Building Blocks

We employ the following tools as our building blocks:

1) **Collision-resistant hash functions**:
A hash function H : {0, 1}∗ → {0, 1}∗ is collision-resistant if the probability of finding two diffenent values x and
y that satisfy H(x) = H(y) is negligible.

2) **Deterministic symmetric encryption**: The encryption algorithm takes a key k and a plaintext m as input, and outputs the ciphertext. We use the notation Enck(m) to denote the encryption algorithm.

3) **Hash-based message authentication codes**:
A hash-based message authentication code HMAC : {0, 1}∗ × {0, 1}∗ → {0, 1}∗ is a deterministic function that takes a key k and an input x, and outputs a value y. We define HMACk(x) def = HMAC(k, x).

4) **Pseudorandom functions**: A pseudorandom function f : {0, 1}∗×{0, 1}∗ → {0, 1}∗ is a deterministic function that takes a key k and a value x, and outputs a value y that is indistinguishable from
a truly random function of the same input x. We define fk(x) def = f(k, x).

5) **Pseudorandom permutations**: A pseudorandom permutation π : {0, 1}∗ × [1, n] → [1, n] is a deterministic function that takes a key k and an integer x where 1 ≤ x ≤ n, and outputs a value y where 1 ≤ y ≤ n that is indistinguishable from a truly random permutation of the same input x. We define πk(x) def = π(k, x).

Algorithm 3
The tag generation algorithm for a leaf node
1: **procedure**
LEAFTAG(αs, kc, αc, c_, i_, li_ , vi_)
2: τ_ ← αsc_
3: ti_ ← fkc (i_kli_kvi_) + αcτ_
4: **return** τ_, ti_
Algorithm 4
The tag generation algorithm for a non-leaf node
1: **procedure** NONLEAFTAG(kc, i, li, vi)
2: τ2i ← t2i − fkc (2ikl2ikv2i)
3: τ2i+1 ← t2i+1 − fkc (2i + 1kl2i+1kv2i+1)
4: **return** ti ← fkc (iklikvi) + τ2i + τ2i+1
5: **Key derivation functions**:
A key derivation function
KDF : {0, 1}∗ × {0, 1}∗ → {0, 1}∗ is a deterministic function that can derive a secret key from two secret values.

4.2 The Pre-process Phase

In the pre-process phase, a user runs the initialization algorithm
(id, e) ← Init(1_, F) which computes:
e ← H(F), id ← H(e).
Then, the user announces that it has a certain file via id. If
the file does not exist, the user goes into the upload phase.
Otherwise, the user goes into the deduplication phase.

4.3 The Upload Phase

Let the file F = (m1, . . . ,mn). The user first invokesthe encoding algorithm (C, T) ← Encode(e, F) which isexecuted as follows.

1) Generate a random key k ← {0, 1}|e|, and computer ← k ⊕ e.

3) Build a HAT from F where the tags in HAT areunassigned.
4) Compute αs ← KDF(k, 1), kc ← KDF(k,
5) Based on the tags of leaf nodes, compute all tags ofnon-leaf nodes in HAT via Algorithm 4.
random key k rather than the hash valuee as the encryption key. Thus, the encoding algorithm isprobabilistic, which is very important to dynamic operations.Note that identical blocks always lead to the sameciphertext, but it is not a security issue because data confidentialityis not the goal of deduplicatable dynamic PoS,and the encryption algorithm in our construction is used forprotecting e.Algorithm 3 is designed for computing the tags of leafnodes. Line 2 randomizes the data block which is used

2) Compute an encryption key ke ← KDF(k, 0). Foreach block m_ (1 ≤ ι ≤ n), compute c_ ←Encke (m_).

6. CONCLUSION

We proposed the comprehensive requirements in multi-usercloud storage systems and introduced the model of deduplicatabledynamic PoS. We designed a novel tool called HATwhich is an efficient authenticated structure. Based on HAT,we proposed the first practical deduplicatable dynamic PoSscheme called DeyPoS and proved its security in the randomoracle model. The theoretical and experimental resultsshow that our DeyPoS implementation is efficient, especially when the file size and the number of the hallenged blocksarelarge.

7.REFERENCES

[1] Kun He, Jing Chen, Ruiying Du, Qianhong Wu, Guoliang Xue, and Xiang Zhang, “DeyPoS: Deduplicatable Dynamic Proof of Storage for Multi-User Environments”, DOI 10.1109/TC.2016.2560812, IEEE Transactions on Computers

[2] Jianghong Wei, Wenfen Liu, Xuexian Hu, “Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption”, Journal Of Latex Class Files, Vol. 14, No. 8, August 2015

[3] R. Di Pietro and A. Sorniotti, “Boosting Efficiency and Security in Proof of Ownership for Deduplication,” in Proc. of ASIACCS, pp. 81–90, 2012.

[4] J. Xu, E.-C. Chang, and J. Zhou, “Weak leakage-resilient clientside deduplication of encrypted data in cloud storage,” in Proc. of ASIACCS, pp. 195–206,2013.

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable data possession at untrusted stores,” in Proc. of CCS, pp. 598–609, 2007.

[6] Q. Zheng and S. Xu, “Secure and efficient proof of storage with deduplication,” in Proc. of CODASPY, pp. 1–12, 2012.

[7] Jingwei Li, Jin Li, Dongqing Xie, and Zhang Cai “Secure Auditing and Deduplicating Data in cloud”, IEEE transaction on computers, vol. 65, no. 8, august 2016