IJERT
ISSN : 2278-0181
International Journal of Engineering Research & Technology

An Approach to Enhance Security in Public Cloud Environment using HCS

P. Abinaya[1]
Dept of Information Technology
E.G.S.Pillay Engineering College
Nagapattinam, Tamilnadu,
India.

P. Pavithra[2]
Dept of Information Technology
E.G.S.Pillay Engineering College
Nagapattinam, Tamilnadu,
India.

P. Suganya[3]
Dept of Information Technology
E.G.S.Pillay Engineering College
Nagapattinam, Tamilnadu,
India.

***Abstract*: Cloud computing is a representation of a movement towards the intensive, large scale specialization. It not only provide convenience and efficiency problems, but also create great challenges in the field of data security and privacy protection. To improve the security of information exchange and storage over public clouds is achieved through encryption and deduplication techniques. To encrypt data and key using hybrid cryptosystems and OpenStack swift algorithm.**

Keywords- Cloud Storage, Data Security, Deduplication, Confidentiality

1 INTRODUCTION

Nowadays, the explosive growth of digital contents continues to rise the demand for new storage and network capacities, along with an increasing need for more cost effective use of storage and network bandwidth for data transfer

As such, the use of remote storage systems is gaining an expanding interest, namely the cloud storage based services, since it provides cost efficient architectures. These architectures support the transmission, storage in a multi-tenant environment, and intensive computation of outsourced data in a pay per use business model.For saving resources consumption in both network bandwidth and storage capacities, many cloud services, namely Dropbox, wuala and Memopal, apply client side deduplication ([5], [10]). This concept avoids the storage of redundant data in cloud servers and reduces network bandwidth consumption associated to transmitting the same contents several times.

Despite these significant advantages in saving resources, client data de duplication brings many security issues, considerably due to the multi owner data possession challenges [10]. For instance, several attacks target either the bandwidth consumption or the confidentiality and the privacy of legitimate cloud users. For example, a user may check whether another user has already uploaded a file, by trying to outsource the same file to the cloud.

Recently, to mitigate these concerns, many efforts have been proposed under different security models ([3],[8], [12], [13], [16]). These schemes are called Proof of Ownership systems (PoW). They allow the storage server check a user data ownership, based on a static and short value (e.g. hash value). These security protocols are designed to guarantee several requirements, namely lightweight of verification and computation efficiency. Even though existing PoW schemes have addressed various security properties, we still need a careful consideration of potential attacks such as Data Leakage and poison attacks, that target privacy preservation and data confidentiality disclosure.

This paper introduces a new cryptographic method for secure Proof of Ownership (PoW), based on the joint use of convergent encryption [15] and the Merkle- [11], for improving data security in cloud storage systems, providing dynamic sharing between users and ensuring efficient data duplication.

Our idea consists in using the Merkle-based Tree over encrypted data, in order to derive a unique identifier of outsourced data. On one hand, this identifier serves to check the availability of the same data in remote cloud servers. On the other hand, it is used to ensure efficient access control in dynamic sharing scenarios.

The remainder of this work is organized as follows. First, Section II describes the state of the art of existing schemes, introducing the general concept of PoW protocols and highlighting their limitations and their security challenges. Then, Section III introduces the system model. Section IV present sour secure PoW scheme and gives a short security analysis. Finally, a performance evaluation is presented, in Section VI before concluding in Section VII.

II. RELATED WORKS AND SECURITY ANALYSIS

The Proof of Ownership (PoW) is introduced by Halevi [8]. It is challenge-response protocol enabling a storage server to check whether a requesting entity is the data owner, based on a short value. That is, when a user wants to upload a data file (D) to the cloud, he first computes and sends a hash value hash= H(D) to the storage server. This latter maintains a database of hash values of all received files, and looks up hash. If there is a match found, then D is already outsourced to cloud servers. As such, the cloud tags the cloud user as an owner of data with no need to upload the file to remote storage servers. If there is no math, then the user has to send the file data (D) to the cloud.

This client side deduplication, referred to as hash-as-aproof[16], presents several security challenges, mainly dueto the trust of cloud users assumption.This Section presents a security analysis of existing PoWschemes.

A. Security Analysis

Despite the significant resource saving advantages, PoW schemes bring several security challenges that may lead tosensitive data.

- Data confidentiality disclosure – hash-as a-proof schemes (e.g. Dropbox) introduce an important data confidentiality concern, mainly due to the static proof client side generation. For instance, if a malicious user has the short hash value of an outsourced data file, he could fool the storage server as an owner trying to upload the requested data file. Then, he gains access to data, by presenting the hash proof. As such, an efficient PoW scheme requires theuse of unpredictable values of verifications.
- Privacy violation – sensitive data leakage is a critical challenge that has not been addressed by Halevi et al.in [8].That is, cloud users should have an efficient way to ensure that remote servers are unable to access outsourced data or to build user profiles._ Poison attack – when a data file D is encrypted on the client side, relying on a randomly chosen encryption key, the cloud server is unable to verify consistency between the uploaded file and the proof value hash.In fact, given a pair (hashD;Enck(D)), the storage server cannot verify, if there is an original data file D, that provides a hash value hash. As such, a malicious user can replace a valid enciphered file with a poisoned file. So, a subsequent user looses his original copy of file, while retrieving the poisoned version.

B. Related Works In 2002, Douceur et al. [4] studied the problem of deduplication in multi-tenant environment. The authors proposed the use of the convergent encryption, i.e., deriving keys from the hash of plaintext. Then, Storer et al. [13] pointed out some security problems, and presented a security model for secure data deduplication. However, these two protocols focus on server-side deduplication and do not consider data leakage settings, against malicious users.

In order to prevent private data leakage, Halevietal. [8] proposed the concept of Proof of Ownership(PoW),while introducing three different constructions, in terms of security and performances. These schemes involve the server challenging the client to present valid sibling paths for a subset of a Merkle tree leaves [11].The first scheme applies erasure coding on the content of the original file. This encoded version is the input for construction of the Merkle tree. The second purpose pre-possesses the data file with a universal hash function instead of erasure coding. The third construction is the most practical approach. Halevi et al. design an efficient hash family, under several security assumptions. Unfortunately, the proof assumes that the data file is sampled from a particular type of distribution. In addition, this construction is given in random oracle model, where SHA256 is considered as a random function. Recently, Ng et al. [12] propose a PoW scheme overencrypted data. That is, the file is divided into fixed-size blocks, where each block has a unique commitment. The hash-tree proof is then built, using the data commitments. Hence, the owner has to prove the possession of a data chunk of a precise commitment, with no need to reveal any secret information. However, this scheme introduces a high computation cost, as requiring generation of all commitments, in every challenging proof request. In [3], the authors presented an efficient PoW scheme.

They use the projection of the file into selected bit-position as a proof of ownership. The main disadvantage of this construction is the privacy violation against honest but curious storage server. In 2013, Jia et al. [16] address the confidentiality preservation concern in cross-user client side deduplication of encrypted data files. They used the convergent encryption approach, for providing deduplication under a weak leakage model.Unfortunately, their paper does not support a malicious storage server adversary.

C. Threat Model

For designing a secure client-side deduplication scheme,we consider two adversaries: malicious cloud user and honest but curious cloud server.

- malicious user adversary – the objective of a malicious user is to convince the cloud server that he is a legitimate data owner. That is, we suppose that the adversary successes to gain knowledge of an arbitrary part of D. This information is then used as a challenging input to the POW protocol.
- curious cloud server adversary – this storage server honestly performs the operations defined by our proposed scheme, but it may actively attempt to gain the knowledge of the outsourced sensitive data. In addition,he may try to build links between user profiles and accessed data files.

III. SYSTEM MODEL

Figure 1 illustrates a descriptive network architecture for cloud storage. It relies on the following entities for the good management of client data:

- Cloud Service Provider (CSP): a CSP has significant resources to govern distributed cloud storage servers and to manage its database servers. It also provides virtual infrastructure to host application services. These services can be used by the client to manage his data stored in the cloud servers.
- Client: a client makes use of provider's resources to store, retrieve and share data with multiple users. A client can be either an individual or an enterprise.
- Users: the users are able to access the content stored inthe cloud, depending on their access rights which are authorizations granted by the client, like the rights to read, write or re-store the modified data in the cloud.These access rights serve to specify several groups of users. Each group is characterized by an identifier IDG and a set of access rights.

In practice, the CSP provides a web interface for the client to store data into a set of cloud servers, which are running in a cooperated and distributed manner. In addition, the

web interface is used by the users to retrieve, modify and restore data from the cloud, depending on their access rights. Moreover, the CSP relies on database servers to map client identities to their stored data identifiers and group identifiers.

IV. NEW INTERACTIVE PROOF OF OWNERSHIP SCHEME

Our secure client-side data deduplication scheme is based on an original use of the convergent encryption [15]. That is, on one hand, when a data owner wants to store a new enciphered data file in remote storage servers
he has first to generate the enciphering key. This data encrypting key is derived by applying a one way hash function on data content.After successfully encrypting the file data, the client

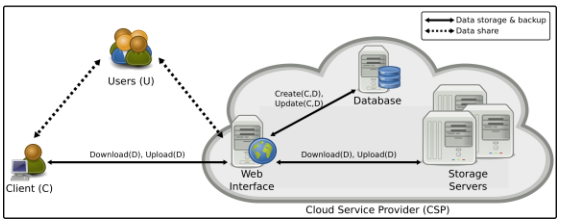

Fig. 1: Architecture of cloud data storage

has to generate the data identifier of enciphered data, in order to check its uniqueness in cloud database, before uploading the claimed file. This data identifier is computed by using a Merkle hash tree, over encrypted contents.Then, for subsequent data outsourcing, the client is not required to send the same encrypted data. However, he has to substitute a client-server interactive proof scheme (PoW), in order to prove his ownership [7].On the other hand, to protect data in public cloud servers from unauthorized entities, the client has to ensure that only authorized users are able to obtain the decrypting keys. As such, the data owner has to encrypt the data deciphering key, using the public key of the recipient user. This key is, then, integrated by the data owner in user metadata, ensuring data confidentiality against malicious users, as well as flexible access control policies. To illustrate our solution for improving data security and efficiency, we first present the different prerequisites and assumptions. Then, we introduce three use cases for storing, retrieving and sharing data among a group of users.

A. Assumptions

Our solution considers the following assumptions. First, we assume that there is an established secure channel between the client and the CSP. This secure channel supports mutual authentication and data confidentiality and integrity. Hence, after successfully authenticating with the CSP, these cloud users share the same resources in a multi-tenant environment.

Second, our solution uses the hash functions in the generation of the enciphering data keys. Hence, we assume that these cryptographic functions are strongly collision resistant, as it is an intractable problem to find the same output for different data files.

B. Prerequisities

Merkle Hash Tree – a Merkle tree MT provides a succinct commitment, called the root value of the Merkle tree, to a data file. That is, the file is divided into blocks, called tree leaves, grouped in pairs and hashed using a collision resistant hash function. The hash values are then grouped in pairs and the process is repeated until the construction of the root value. The Merkle tree proof protocol requires that the prover has the original data file. That is, the verifier chooses a number of leaf indexes and asks the verifier to provide the corresponding leaves. As such, the verifier has to send these leaves with a sibling valid path.

Interactive Proof System [7] – this proof system is an interactive game between two parties: a challenger and a verifier that interact in a common input, satisfying the correctness properties (i.e. completeness and soundness).

In the following, we introduce our client-side deduplication construction, based on three different scenarios: storage, backup and sharing schemes.

C. Cloud Data Storage

When a client wants to store a new data file f in the cloud, he derives the enciphering key kf from the data contents, based on a one-way hash function H(). Note that data are stored enciphered in cloud servers, based on a symmetric algorithm. Hence, the data owner has to encipher the data file that he intends to outsource. Then, he generates the data identifier MTf . That is, it is the Merkle Tree over encrypted data. This identifier, associated to the file, must be unique in the CSP database. Thus, the client starts the storage process by sending a ClientRequestVerif message to verify the uniqueness of the generated MTf to his CSP.

*1) New Data File Storage:*The storage process consists in exchanging the four following messages:

- ClientRequestVerif : this first message contains the generated data identifier MTf , associated to a nonce n. Note that the nonce is used to prevent from replay attack or potential capture of the data identifier. This message is a request for the verification of the uniqueness of the MTf . The CSP replies with a ResponseVerif message to validate or unvalidate the claimed identifier. Note that if the sent identifier exists, the client has to perform a subsequent upload extra-proof procedure with the provider (cf, Section IV-C2). Once the verification holds, the cloud server asks the client to send only the access rights of authorized users.
- ResponseVerif:this acknowledgement message is generated by the CSP to inform the client about the existence of the requested MTf in its database.
- ClientRequestStorage: this message is sent by the client. If the file does not exist in the cloud servers, the client sends the file that he intends to store in the cloud, and the data decrypting key kf enciphered with the public keys of authorized users. Then, the enciphered kf is included in the meta data of the file and it serves as an access rights provision.
- ResponseStorage: this acknowledgement message, sent by the CSP, is used to confirm to the client the success of his data storage.This message contains the Uniform Resource Identifier(URI) of the outsourced data.

*2) Subsequent Data File Storage:*When a client wants to store a previous outsourced data file, he sends the data file identifier MTf to the cloud provider. Since the claimed identifier in cloud database, the cloud has to verify that the requesting entity is a legitimate client. That is, the subsequent data storage procedure include these four messages:

- ClientRequestVerif: a subsequent data owner includes in this first message the generated data identifier MTf , associated to a nonce n, in order to check its uniqueness in cloud database.
- OwnershipRequest: this message is sent by the CSP, to verify the client's data ownership. It contains random leaves' indices of the associated Merkle tree of the requested file. Upon receiving this message, the client has to compute the associated sibling path of each leaf, based on the stored Merkle tree, in order to prove his ownership of the requested file.
- ClientResponseOwnership: in his response, the client must include a valid sibling path of each selected leaf. The CSP verifies the correctness of the paths provided by the client. We must note that this data subsequent storage process stops if the verification fails.
- ResponseStorage: if the data ownership verification holds, the CSP sends an acknowledgement, to confirm the success of storage, while including the URI of the requested data.

D. Cloud Data Backup

The data backup process starts when the client requests for retrieving the data previously stored in the cloud. The data backup process includes the following messages:

- ClientRequestBackup: it contains the URI of the requested data that the client wants to retrieve. Upon receiving this client request, the CSP verifies the client ownership of the claimed file and generates a ResponseBackup message.
- ResponseBackup: in his response, the CSP includes the encrypted outsourced data kf (f). Upon receiving the ResponseBackup message, the client first retrieve the file metadata and deciphers the data decrypting key kf , using his secret key. Then, he uses the derived key to decrypt the request data file.

E. Cloud Data Sharing

We consider the data sharing process, where the client outsources his data to the cloud and authorizes a group of users to access the data. Next, we refer to these user(s) as the recipient(s) and to the data owner as the depositor. We must note that our proposal does not require the recipients to be connected during the sharing process. Indeed, recipients' access rights are granted by the data owner and managed by the CSP. That is, these access rights are also included in the meta data file. In addition, the CSP is in charge of verifying each recipient access permissions before sending him the outsourced data.In practice, each recipient is assumed to know the URI of the outsourced data. This URI distribution problem can be solved in two ways. Either the depositor sends the URI to the recipient as soon as he stores data or a proxy is in charge of distributing the URIs. Once the depositor stored the data with the authorized access rights of the group, each member of the group can start the data sharing process based on the two following messages:

- UserRequestAccess: This message contains the URI of the requested file. When receiving this message, the CSP searches for the read/write permissions of the recipient, and then, he generates a Response Access message.
- ResponseAccess: the CSP includes, in its response, the enciphered file kf (f). Upon receiving this message,each recipient retrieves the data decrypting key from user metadata. That is, he deciphers the associated symmetric key with his own private key. Then, he performs a symmetric decryption algorithm to retrieve the plaintext. Our proposal provides a strong solution to improve the confidentiality of data in the cloud. In addition, the access to outsourced data is controlled by two processes. First, there is a traditional access list managed by the CSP. Second, the client has to own the private decrypting key to get the secret needed to retrieve the symmetric key compulsory needed to decipher data.

V. CONCLUSION

The growing need for secure cloud storage services and the attractive properties of the convergent cryptography lead us to combine them, thus, defining an innovative solution to the data outsourcing security and efficiency issues.

Our solution is based on a cryptographic usage of symmetric encryption used for enciphering the data file and asymmetric encryption for key, due to the highest sensibility of these information towards several intrusions. In addition, thanks to the Merkle tree properties, this proposal is shown to support data deduplication, as it employs an pre-verfication of data existence, in cloud servers, which is useful for saving bandwidth. Besides, our solution is also shown to be resistant to unauthorized access to data and to any data disclosure during sharing process, providing two levels of access control verification. Finally, we believe that cloud data storage security is still full of challenges and of paramount importance, and many research problems remain to be identified.

REFERENCES

[1] https://github.com/openstack/swift.
[2] L. Ben. On the implementation of pairing-based cryptosystems, 2007.
[3] R. Di Pietro and A. Sorniotti. Boosting efficiency and security in proof of ownership for deduplication. In Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ASIACCS '12, pages 81–82, New York, NY, USA, 2012. ACM.
[4] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In In Proceedings of 22nd International Conference on Distributed Computing Systems (ICDCS, 2002.
[5] M. Dutch. Understanding data deduplication ratios. SNIA White Paper, June 2008.
[6] T. G. et al. GNU multiple precision arithmetic library 4.1.2, December 2002.
[7] O. Goldreich. Foundations of Cryptography: Basic Tools. Cambridge University Press, New York, NY, USA, 2000.

[8] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In Proceedings of the 18th ACM conference on Computer and communications security, CCS '11, pages 491–500, New York, NY, USA, 2011. ACM.

[9] D. Hankerson, A. J. Menezes, and S. Vanstone. Guide to Elliptic Curve Cryptography. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.