ISSN:0975-9646

Differential Privilege based Secure Authorized Deduplication Using Public Private Cloud

Amit Harish Palange , Prof.Deepak Gupta
Computer Department, Savitribai Phule Pune University, Pune India

***Abstract*—To reduce the amount of storage space and save bandwidth in cloud storage various Data compression techniques are used, Data deduplication is one of the important data compression technique. Data deduplication removes the duplicate copies of repeating data. To encrypt the sensitive data, the convergent encryption technique has been proposed. It is first formal attempt to address the problem of secure authorized data deduplication. Data and the differential privileges of users are considered while duplicate check, thus differing from traditional deduplication systems. Several new deduplication constructions supporting authorized duplicate check in multi cloud architecture are presented here. Security analysis shows that proposed system is secure in terms of definitions specified in the proposed security model. Prototype of proposed authorized duplicate check is implemented and test bed experiments conducted using this prototype. The result shows that proposed authorized duplicate check scheme incurs minimal overhead as compared to normal operations.**

***Keywords*—Differential privileges ,Secure Deduplication, authorized duplicate check, confidentiality.**

I. INTRODUCTION

Cloud computing is a climbing engineering that as of late has drawn essential consideration from each one exchange and academe. It gives benefits over the web, by abuse Cloud computing client will use the net administrations of different bundle as opposed to purchasing or putting in them all customers are further considered in duplicate check other than the data itself. We also demonstrate a couple of new deduplication advancements supporting sanction duplicate make a case a cream cloud building configuration. Security analyzation displays that our arrangement is secure with respect to the definitions characterized in the proposed security model.

There is huge amount of data stored in cloud, deduplication is useful and efficient technique to make data management more scalable. Data deduplication is a specialized data compression technique for removing duplicate copies of repeating data in storage. Deduplication technique is useful to improve storage utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent. There will be no need to keep multiple data copies which has same content, redundant data is eliminated using Deduplication and only one physical copy is stored and when redundant data is referred it is directed to that physical copy. Deduplication takes place at two levels namely at file level and block level. Duplicate copies of the same file are eliminated at file level deduplication. At the block level, duplicate blocks of data that occur in non-identical files are eliminated. Data Deduplication has several benefits, but still users' sensitive data are susceptible to both insider and outsider attacks which causes privacy and security concerns. Though traditional encryption provides confidentiality it is not compatible with deduplication. In, traditional encryption different users use their own key to encrypt the data. Thus, different ciphertexts are created from identical data copies which will make deduplication impossible. To make deduplication feasible Convergent encryption [8] has been proposed to enforce data confidentiality. Convergent key is used to encrypt/decrypt a data copy. Convergent key is obtained by computing the cryptographic hash value of the content of the data copy. Users retain the keys after generating it, and send the ciphertext to the cloud. Since the encryption operation is deterministic and is derived from the data content, identical data copies will generate the same convergent key and hence the same ciphertext. A secure proof of ownership protocol is used to prevent unauthorized access and to provide the proof that the user owns the same file when a duplicate is found. After the proof, subsequent users with the same file will be provided a pointer from the server without needing to upload the same file. A user can download the encrypted file with the pointer from the server, which can only be decrypted by the corresponding data owners with their convergent keys. Thus, to preform deduplication on cipher texts in cloud, convergent encryption is needed and the proof of ownership prevents the unauthorized user to access the file.

II. PRELIMINARIES

We first define the notations used in this paper, review some secure primitives used in our secure deduplication.

A. *Symmetric encryption:*

A common secret key κ is used to encrypt and decrypt information in Symmetric encryption. A symmetric encryption scheme consists of three primitive functions:

- KeyGenSE(1_) -> κ is the key generation algorithm that generates κ using security parameter 1_;
- EncSE(κ,M) -> C is the symmetric encryption algorithm that takes the secret κ and message M and then outputs the ciphertext C; and
- DecSE(κ,C) ->M is the symmetric decryption algorithm that takes the secret κ and ciphertext C and then outputs the original message M.

B. *Convergent encryption:*

Convergent encryption [4], [8] provides data confidentiality in deduplication. A user (or data owner) derives a convergent key from each original data copy and encrypts the data copy with the convergent key. From this data copy a tag is also derived and this tag is then used to detect the duplicate. It is assumed that the tag correctness property [4] holds, i.e., if two data copies are the same, then the tags derived from them is also same. To detect duplicates, to check if the identical copy has been already stored the user first sends the tag to the server side. Note that both the convergent key and the tag are independent i.e there is no relation between this two, the tag cannot be used to deduce the convergent key and compromise data confidentiality. Both the encrypted data copy and its corresponding tag will be stored on the server side.

C. *Proof of ownership:*

The notion of proof of ownership (PoW) enables users to prove their ownership of data copies to the storage server. PoW is implemented as an interactive algorithm (denoted by PoW) run by a prover (i.e., user) and a verifier (i.e., storage server). A short value $\phi(M)$ is derived from a data copy M by verifier. To prove the ownership of the data copy M, the prover needs to send ϕ' to the verifier such that $\phi' = \phi(M)$. The formal security definition for PoW roughly follows the threat model in a content distribution network, where an attacker does not know the entire file, but has accomplices who have the file. The accomplices follow the "bounded retrieval model", such that they can help the attacker obtain the file, subject to the constraint that they must send fewer bits than the initial min-entropy of the file to the attacker.

D. *Identification Protocol:*

An identification protocol can be described with two phases: Proof and Verify. In the stage of Proof, a prover/user U can demonstrate his identity to a verifier by performing some identification proof related to his identity. The input of the prover/user is his private key skU that is sensitive information such as private key of a public key in his certificate or credit card number etc. that he wouldn't share with the other users. The verifier performs the verification with input of public information pkU related to skU. At the conclusion of the protocol, the verifier outputs either accept or reject to denote whether the proof is passed or not. There are many efficient identification protocols in literature, including certificate-based, identity-based identification etc.

III. System Model

A. *Hybrid Architecture for Secure Deduplication*

At a high level, our setting of interest is an enterprise network, which consist of a group of affiliated clients (for example, employees of a company) who will use the S-CSP and store data with deduplication technique. In this setting, deduplication can be frequently used in these settings for data backup and disaster recovery applications while greatly reducing storage space. Such systems are widespread and are often more suitable to user file backup and synchronization applications than richer storage abstractions. Users have access to the private cloud server, a semitrusted third party which will aid in performing deduplicable encryption by generating file tokens for the requesting users. The role of the private cloud server is further described below. Users are also provisioned with per-user encryption keys and credentials (e.g., user certificates). In this paper, we will only consider the file level deduplication for simplicity. In another word, we refer a data copy to be a whole file and file-level deduplication which eliminates the storage of any redundant files. Actually, block-level deduplication can be easily deduced from file-level deduplication. To upload a file, a user first performs the file-level duplicate check. If the file is a duplicate, then all of the blocks of the file must be duplicates as well; If it's not duplicate then, the user further performs the block-level duplicate check and identifies the unique blocks to be uploaded. Each data copy (i.e., a file or a block) is associated with a token for the duplicate check.

B .*Adversary Model*

Typically, we assume that the public cloud and private cloud are both "honest-but-curious". Specifically they will follow our proposed protocol, but try to find out as much secret information as possible based on their possessions. Users would try to access data either within or out of the scopes of their privileges.

In this paper, we suppose that all the files are sensitive and needed to be fully protected against both public cloud and private cloud. Under the assumption, two kinds of adversaries are considered, that is, 1) external adversaries which aim to extract secret information as much as possible from both public cloud and private cloud; 2) internal adversaries who aim to obtain more information on the file from the public cloud and duplicate-check token information from the private cloud outside of their scopes. Such adversaries may include S-CSP, private cloud server and authorized users. The detailed security definitions against these adversaries are discussed below and in Section 5, where attacks launched by external adversaries are viewed as special attacks from internal adversaries.

C. Design Goals

In this paper, we address the problem of privacy preserving deduplication in cloud computing and propose a new deduplication system supporting for

- *Differential Authorization.* Each authorized user is able to get his/her individual token of his file to perform duplicate check based on his privileges. Under this assumption, any user cannot generate a token for duplicate check out of his privileges or without the help of the private cloud server.
- *Authorized Duplicate Check.* Authorized user is able to use his/her individual private keys to generate query for certain file that he needs and the privileges he/she owned with the help of private cloud, while the public cloud performs duplicate check directly and tells the user if there is any duplicate.

The security requirements considered in this paper lie in two folds, including the security of file token and security of data files. For the security of file token, two aspects are defined as unforgeability and in distinguishability of file token.

IV. SECURE DEDUPLICATION SYSTEMS

Main Idea is to support authorized deduplication, the tag of a file F will be computed using the file F itself and the privilege. To be different from the traditional notation of tag, we refer to it as a file token instead. For supporting authorized access, a secret key kp will be bounded with a privilege p to generate a file token.

Let $\phi'_{F,p}$ = TagGen(F, k_p) denote the token of F that is only allowed to access by user with privilege p. In another word, the token $\phi'_{F,p}$ could only be computed by the users with privilege p. As a result, if a file has been uploaded by a user with a duplicate token $\phi'_{F,p}$, then a duplicate check sent from another user will be successful if and only if he also has the file F and privilege p. Such a token generation function could be easily implemented as H(F, k_p), cryptographic hash function is denoted by H(·) .

A. *A First Attempt*

Before introducing our construction of differential deduplication, this is a straightforward attempt with the technique of token generation TagGen(F, k_p) above to design a required deduplication system. The idea of this basic construction is to issue corresponding privilege keys to each user, these users will compute the file tokens and perform the duplicate check based on the privilege keys and files. In more details, suppose that there are N users in the system and the privileges in the universe is defined as $P = \{fp_1, \dots, fp_s\}$. For each privilege fp in P, a private key k_p will be selected. For a user U with a set of privileges P_U, he will be assigned the set of keys $\{k_{pi}\}_{pi \in PU}$.

B. *Our Proposed System Description*

To solve the problems of the construction in Section 4.1, we propose another advanced deduplication system supporting authorized duplicate check. In this new deduplication system, hybrid cloud architecture is introduced to solve the problem. The private keys for privileges will not be issued to users directly, which will be kept and managed by the private cloud server instead. In this way, the users will not be able to share these private keys of privileges in our proposed construction, which means that it can prevent the privilege key sharing among users in the above straightforward construction. If user needs a file token, then he needs to send a request to the private cloud server. The intuition of this construction can be described as following manner. If user wants to perform the duplicate check for some file, the he must get the file token from the private cloud server. Then the private cloud server will also check the user's identity and only then corresponding file token will be issued to the user. Before uploading the file the authorized duplicate check for this file can be performed by the user with the public cloud. It depends on the results of duplicate check, the user either runs PoW or uploads this file.

C. *Further Enhancement*

Differential privilege duplicate is supported by the above solution, it is inherently subject to bruteforce attacks launched by the public cloud server, which can recover files falling into a known set. More specifically, knowing that the target file space underlying a given ciphertext C is drawn from a message space
$S = \{F_1, \dots, F_n\}$ of size n, the public cloud server can recover F after at most n off-line encryptions. That is, for each $i = 1, \dots, n$, it simply encrypts F_i to get a ciphertext denoted by C_i. If $C = C_i$, it means that the underlying file is F_i. Security is thus only possible when such a message is unpredictable. This traditional convergent encryption will be insecure for predictable file.

V. SECURITY ANALYSIS

Our system is designed to solve the differential privilege problem in secure deduplication. The security will be analyzed in terms of two aspects. First the authorization of duplicate check and the second is confidentiality of data. Some basic tools have been used to construct the secure deduplication, which are assumed to be secure. These basic tools include the symmetric encryption scheme, convergent encryption scheme, and the PoW scheme. Based on this assumption, we can show that these systems are secure with respect to the following security analysis.

A. *Security of Duplicate-Check Token*

We consider several types of privacy we need to protect, as follows i) unforgeability of duplicate-check token: There are two types of adversaries, first is external adversary and the other one is internal adversary. The external adversary can be viewed as an internal adversary without any privilege. If a user has privilege p, then it is mandatory to ensure that the adversary cannot forge and output a valid duplicate token with any other privilege p' on any file F, where p does not match p'. Furthermore, it also requires that if the adversary does not make a request of token with its own privilege from private cloud server, then it is also not possible to forge and output a valid duplicate token with p on any F that has been queried. Comparing the attacking power of these two adversaries we found that, the internal adversaries have more attack power than the external adversaries and thus we only need to consider the security against the internal attacker that is , ii) indistinguishability of duplicatecheck token : this property is also defined in terms of two aspects as the definition of unforgeability. First, if a user has privilege p, given a token ϕ', it requires that the adversary cannot distinguish which privilege or file in the token if p does not match p'. Furthermore, it also require that if the adversary does not make a request of token with its own privilege from private cloud server, it cannot distinguish a valid duplicate token with p on any other F that the adversary has not queried. In the security definition of indistinguishablity, we require that the adversary is not allowed to collude with the public cloud servers. Actually, such an assumption could be removed if the private cloud server maintains the tag list for all the files uploaded. Similar to the analysis of unforgeability, the security against external adversaries is implied in the security against the internal adversaries.

B. Confidentiality of Data

The data will be encrypted in our deduplication system before outsourcing to the S-CSP. Two kinds of different encryption methods have been applied in our two constructions. Thus, we will analyze them respectively. In the scheme in Section 4.2, the data is encrypted with the traditional encryption scheme. The data encrypted with such encryption method cannot achieve semantic security as it is inherently subject to brute force attacks that can recover files falling into a known set. Thus, several new security notations of privacy against chosen-distribution attacks have been defined for unpredictable message. In another word, the adapted security definition guarantees that the encryptions of two unpredictable messages should be indistinguishable. Thus, the security of data in our first construction could be guaranteed under this security notion.

VI . IMPLEMENTATION

We implement a prototype of the proposed authorized deduplication system addressing the differential privilege problem, in which we model three entities as separate Java programs. A *Client* program is used to model the data users to carry out the file upload process. A *Private Server* program is used to model the private cloud which manages the private keys and handles the file token computation. A *Storage Server* program is used to model the S-CSP which stores and deduplicates files. We implement cryptographic operations of hashing and encryption with the Open SSL library [1].We also implement the communication between the entities based on HTTP, using GNU Libmicrohttpd and libcurl . Thus, users can issue HTTP Post requests to the servers.

VII. CONCLUSION

In this paper, the notion of authorized data deduplication was proposed to protect the data security by including differential privileges of users in the duplicate check. We have also presented several new deduplication constructions supporting authorized duplicate check with the help of Public and Private cloud architecture, in this scheme the duplicate-check tokens of files are generated by the private cloud server with the help of private keys. Furthermore, our Security analysis demonstrates that our schemes are secure in terms of insider and outsider attacks specified in the proposed security model. To give proof of this concept, we implemented a prototype of our proposed authorized duplicate check scheme and conduct testbed experiments on our prototype. We have also showed that our authorized duplicate check scheme is more efficient and incurs minimal overhead compared to convergent encryption and network transfer.

REFERENCES

[1] OpenSSL Project. http://www.openssl.org/.

[2] P. Anderson and L. Zhang. Fast and secure laptop backups with encrypted de-duplication. In *Proc. of USENIX LISA*, 2010.

[3] M. Bellare, S. Keelveedhi, and T. Ristenpart. Dupless: Serveraided encryption for deduplicated storage. In *USENIX Security Symposium*, 2013.

[4] M. Bellare, S. Keelveedhi, and T. Ristenpart. Message-locked encryption and secure deduplication. In *EUROCRYPT*, pages 296–312, 2013.

[5] M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. *J. Cryptology*, 22(1):1–61, 2009.

[6] M. Bellare and A. Palacio. Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *CRYPTO*, pages 162–177, 2002.

[7] S. Bugiel, S. Nurnberger, A. Sadeghi, and T. Schneider. Twin clouds: An architecture for secure cloud computing. In *Workshop on Cryptography and Security in Clouds (WCSC 2011)*, 2011.

[8] 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 *ICDCS*, pages 617–624, 2002.\