International Journal of Research in Advent Technology, Special Issue, March 2019
E-ISSN: 2321-9637
International Conference on Technological Emerging Challenges (ICTEC-2019)
Available online at www.ijrat.org

A Hybrid Cloud Approach For Secure Authorized Deduplication

[1]B.v.Suresh Reddy,[2]P.lakshmi,[3]U.sirisha,[4]R.Bharathi,[5]P.Pranay
Asst.Prof,CSE,Tirumala Engineering College,Nararaopet,India
B.Tech Students , Tirumala Engineering College , Narasaraopet , India.
1.pynamlakshmi1998@gmail.com,2.sirishaupputuri1998@gmail.com
3.ravuri911@gmail.com,4.pranaychowdary76@gmail.com

Abstract : Data deduplication is one of important data compression techniques for eliminating duplicate copies of repeating data And has been widely used in cloud storage to reduce the amount of storage space and save bandwidth. To protect the confidentialityof sensitive data while supporting deduplication, the convergent encryption technique has been proposed to encrypt the data beforeoutsourcing. To better protect data security, this paper makes the first attempt to formally address the problem of authorized data deduplication. Different from traditional deduplication systems, the differential privileges of users are further considered in duplicatecheck besides the data itself.We also present several new deduplication constructions supporting authorized duplicate check in a hybrid cloud architecture. Security analysis demonstrates that our scheme is secure in terms of the definitions specified in the proposed security model. As a proof of concept, we implement a prototype of our proposed authorized duplicate check scheme and conduct testbed experiments using our prototype. We show that our proposed authorized duplicate check scheme incurs minimal overhead compared to normal operations.

1. INTRODUCTION

Cloud computing provides seemingly unlimited "virtualized resources to users as services across the whole Internet,while hiding platform and implementation details.Today's cloud service providers offer both highly available storage and massively parallel computing resources at relatively low costs. To make data management scalable in cloud computing,deduplication has been a well-known technique and has attracted more and more attention recently.Data deduplication is a specialized data compression technique for eliminating duplicate copies of repeating data in storage.The technique is used to improve storage utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent. Instead of keeping multiple data copies with the same content, deduplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. Deduplication can take place at either the file level or the block level. For filelevel deduplication, it eliminates duplicate copies of the same file. Deduplication can also take place at the block level, which eliminates duplicate blocks of data that occur in non-identical files. Although data deduplication brings a lot of benefits,security and privacy concerns arise as users' sensitive data are susceptible to both insider and outsider attacks.Traditional encryption, while providing data confidentiality,is incompatible with data deduplication. Specifically,traditional encryption requires different users toencrypt their data with their own keys. Thus, identical data copies of different users will lead to different ciphertexts,making deduplication impossible. Convergent encryption [8] has been proposed to enforce data confidentialitywhile making deduplication feasible. It encryptsdecrypts a data copy with a *convergent key*, whichis obtained by computing the cryptographic hash value of the content of the data copy. After key generation and data encryption, users retain the keys 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. To prevent unauthorized access, a secure proof of ownership protocol [11] is also needed to provide the proof that the user indeed 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, convergent encryption allows the cloud to perform deduplication on the ciphertexts and the proof of ownership prevents the unauthorized user to access the file. However, previous deduplication systems cannot support differential authorization duplicate check, which is importantin many applications. In such an authorized deduplication system, each user is issued a set of privileges during system initialization (in Section 3, we elaborate the definition of a privilege with examples). Each file uploaded to the cloud is also bounded by a set of privileges to specify which kind of users is allowed to perform the duplicate

check and access the files. Before submitting his duplicate check request for some file, the user needs to take this file and his own privileges as inputs. The user is able to find a duplicate for this file if and only if there is a copy of this file and a matched privilege stored in cloud. For example, in a company, many different privileges will be assigned to employees.In order to save cost and efficiently management, the data will be moved to the storage server provider (SCSP) in the public cloud with specified privileges and the deduplication technique will be applied to store only one copy of the same file. Because of privacy consideration,some files will be encrypted and allowed the duplicate check by employees with specified privileges to realize the access control. Traditional deduplication systems based on convergent encryption, although providing Other words, no differential privileges have been considered in the deduplication based on convergent encryption technique. It seems to be contradicted if we want to realize both deduplication and differential authorization duplicate check at the same time.

A. Objective Of The Project

In this paper, aiming at efficiently solving the problem of deduplication with differential privileges in cloud computing, we consider a hybrid cloud architecture consisting of a public cloud and a private cloud. Unlike existing data deduplication systems, the private cloud is involved as a proxy to allow data owner/users to securely perform duplicate check with differential privileges. Such an architecture is practical and has attracted much attention from researchers. The data owners only outsource their
data storage by utilizing public cloud while the data operation is managed in private cloud. A new deduplication system supporting differential duplicate check is proposed under this hybrid cloud architecture where the S-CSP resides in the public cloud. The user is only allowed to perform the duplicate check for files marked with the corresponding privileges. Furthermore, we enhance our system in security. Specifically, we present an advanced scheme to support stronger security by encrypting the file with differential privilege keys. In this way, the users without corresponding privileges cannot perform the duplicate check. Furthermore, such unauthorized users cannot decrypt the ciphertext even collude with the S-CSP. Finally, we implement a prototype of the proposed authorized duplicate check and conduct testbed experiments to evaluate the overhead of the prototype. We show that the overhead is minimal compared to the normal convergent encryption and file upload operations.

2. EXISTING SYSTEM:

Data deduplication systems, the private cloud is involved as a proxy to allow data owner/users to securely perform duplicate check with differential privileges.Such architecture is practical and has attracted much attention from researchers. The data owners only outsource their data storage by utilizing public cloud while the data operation is managed in private cloud.

Disadvantages Of Existing System:

1. Traditional encryption, while providing data confidentiality, is incompatible with data deduplication.
2. Identical data copies of different users will lead to different ciphertexts, making deduplication impossible.

3. PROPOSED SYSTEM

In this paper, we enhance our system in security. Specifically, we present an advanced scheme to support stronger security by encrypting the file with differential privilege keys. In this way, the users without corresponding privileges cannot perform the duplicate check. Furthermore, such unauthorized users cannot decrypt the cipher text even collude with the S-CSP. Security analysis demonstrates that our system is secure in terms of the definitions specified in the proposed security model.

Advantages of Proposed System:

1. The user is only allowed to perform the duplicate check for files marked with the corresponding privileges.
2. We present an advanced scheme to support stronger security by encrypting the file with differential privilege keys.
3. Reduce the storage size of the tags for integrity check. To enhance the security of deduplication and protect the data confidentiality.

4. IMPLEMENTATION:

We implement a prototype of the proposed authorized deduplication system, in which we model three entities as separate C++ 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 OpenSSL library [1].We also implement the communication between the entities based on HTTP, using GNU Libmicrohttpd [10] and libcurl [13].Thus, users can issue HTTP Post requests to the servers.

Our implementation of the Client provides the following function calls to support token generation and

deduplication along the file upload process.

FileTag(File) - It computes SHA-1 hash of the File as File Tag;

TokenReq(Tag, UserID) - It requests the Private Server for File Token generation with the File Tag and User ID;

DupCheckReq(Token) - It requests the Storage Server for Duplicate Check of the File by sending the file token received from private server;

ShareTokenReq(Tag, {Priv.}) - It requests the Private Server to generate the Share File Token with

the File Tag and Target Sharing Privilege Set;

FileEncrypt(File) - It encrypts the File with Convergent Encryption using 256-bit AES algorithm in cipher block chaining (CBC) mode, where the convergent key is from SHA-256 Hashing of t he file and

FileUploadReq(FileID, File, Token) – It uploads the File Data to the Storage Server if the file is Unique and updates the File Token stored. Our implementation of the Private Server includes corresponding request handlers for the token generation and maintains a key storage with Hash Map.

TokenGen(Tag, UserID) - It loads the associated privilege keys of the user and generate the token with HMAC-SHA-1 algorithm; and

ShareTokenGen(Tag, {Priv.}) - It generates the share token with the corresponding privilege keys of the sharing privilege set with HMAC-SHA-1 algorithm.Our implementation of the **Storage Server** provides deduplication and data storage with following handlers and maintains a map between existing files and associated token with Hash Map.

DupCheck(Token) - It searches the File to Token Map for Duplicate; and

FileStore(FileID, File, Token) - It stores the File on Disk and updates the Mapping.

9

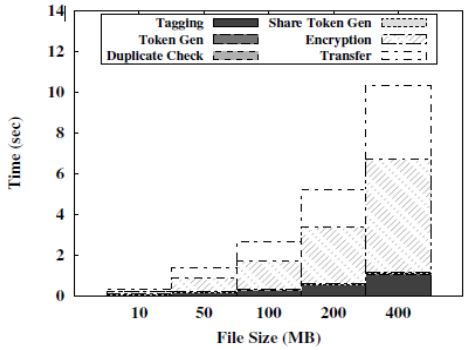


Fig. 2. Time Breakdown for Different File Size

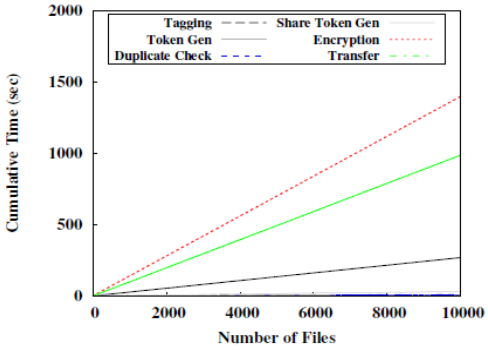


Fig. 3. Time Breakdown for Different Number of Stored Files

5. EVALUATION:

We conduct testbed evaluation on our prototype. Our evaluation focuses on comparing the overhead induced by authorization steps, including file token generation and share token generation, against the convergent encryption and file upload steps. We evaluate the overhead by varying different factors, including 1) File Size 2) Number of Stored Files 3) Deduplication Ratio 4) Privilege Set Size . We also evaluate the prototype with a real-world workload based on VM images.

File Size:

To evaluate the effect of file size to the time spent on different steps, we upload 100 unique files of particular file size and record the time break down. Using the unique files enables us to evaluate the worst-case scenario where we have to upload all file data. The average time of the steps from test sets of different file size are plotted in Figure. In contrast, other steps such as token generation and duplicate check only use the file metadata for computation and therefore the time spent remains constant.With the file size increasing from 10MB to 400MB, the overhead of the proposed authorization steps decreases from 14.9% to 0.483%.

Number of Stored Files:

To evaluate the effect of number of stored files in the system, we upload 10000 10MB unique files to the system and record the breakdown for every file upload. From Figure 3, every step remains constant along the time.Token checking is done with a hash table and a linear search would be carried out in case of collision. Despite of the possibility of a linear search, the time taken in duplicate check remains stable due to the low collision probability.

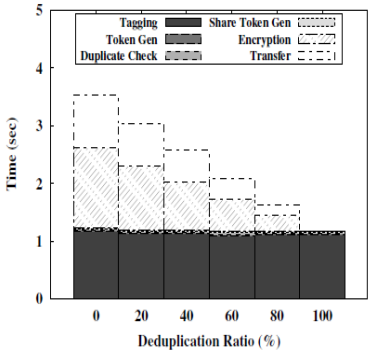
Deduplication Ratio:

To evaluate the effect of the deduplication ratio, we prepare two unique data sets, each of which consists of 50 100MB files. We first upload the first set as an initial upload. For the second upload, we pick a portion of 50 files, according to the given deduplication ratio, from the initial set as duplicate files and remaining files from the second set as

unique files.The time spent on duplicate check also decreases as the searching would be ended when duplicate is found. Total time spent on uploading the file with deduplication ratio at 100% is only 33.5% with unique files.

Privilege Set Size:

To evaluate the effect of privilege set size, we upload 100 10MB unique files with different size of the data owner and target share privilege set size. In Figure 5, it shows the time taken in token generation increases linearly as more keys are associated with the file and also the duplicate check time. While the number of keys increases 100 times from 1000 to 100000, the total time spent only increases to 3.81 times and it is noted that the file size of the experiment is set at a small level (10MB),the effect would become less significant in case largefiles.



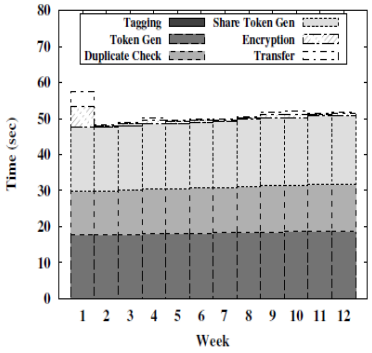


Fig. 4. Time Breakdown for Different Deduplication Ratio Fig. 6. Time Breakdown for the VM dataset.

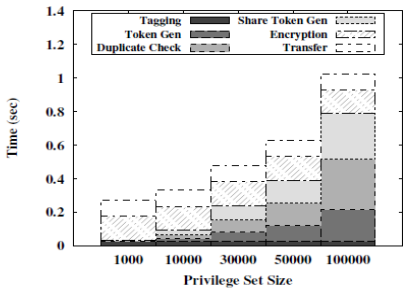


Fig. 5. Time Breakdown for Different Privilege Set Size

6. SYSTEM ARCHITECTURE:

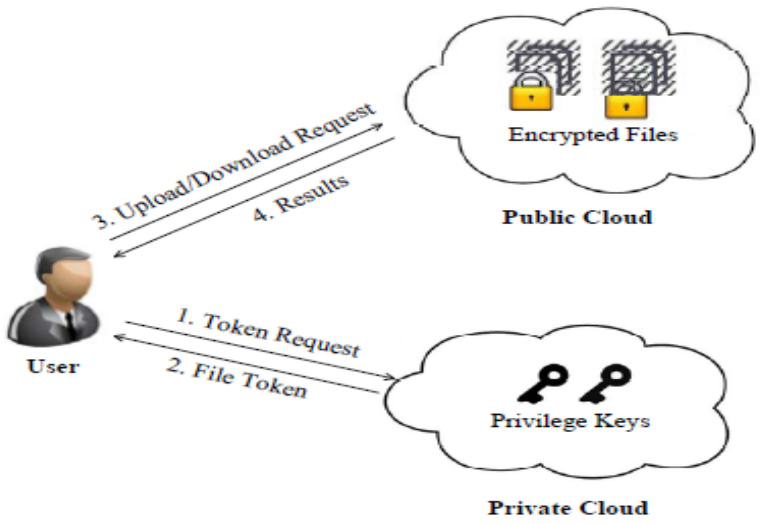


Fig. 1. Architecture for Authorized Deduplication

7. 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, that is, the authorization of duplicate check and the 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 convergent encryption scheme,symmetric encryption scheme, and the PoW scheme.Based on this assumption, we show that systems are secure with respect to the following security analysis.

Confidentiality of Data:

The data will be encrypted in our deduplication system before outsourcing to the S-CSP. Furthermore, two kinds of different encryption methods have been applied in our two constructions. Thus, we will analyze them respectively The data encrypted with such encryption method cannot achieve semantic security as it is inherently subject to bruteforce 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. The security analysis for external adversaries and internal adversaries is almost identical, except the internal adversaries are provided with some convergent encryption keys additionally. However, these convergent encryption keys have no security impact on the data confidentiality because these convergent encryption keys are computed with different privileges. Recall that the data are encrypted with the symmetric key encryption technique, instead of the convergent encryption method. Though the symmetric key *k* is randomly chosen, it is encrypted by another convergent encryption key *kF;p*. Thus, we still need analyze the confidentiality of data by considering the convergent encryption.

8. 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 also presented several new deduplication constructions supporting authorized duplicate check in hybrid cloud architecture, in which the duplicate-check tokens of files are generated by the private cloud server with private keys. Security analysis demonstrates that our schemes are secure in terms of insider and outsider attacks specified in the proposed security model. As a proof of concept, we implemented a prototype of our proposed authorized duplicate check scheme and conduct testbed experiments on our prototype. We showed that our authorized duplicate check scheme 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 *Workshopon 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.
[9] D. Ferraiolo and R. Kuhn. Role-based access controls. In *15th NIST-NCSC National Computer Security Conf.*, 1992.
[10] GNU Libmicrohttpd. http://www.gnu.org/software/libmicrohttpd/.
[11] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of ownership in remote storage systems. In Y. Chen, G. Danezis, and V. Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 491–500. ACM, 2011.
[12] J. Li, X. Chen, M. Li, J. Li, P. Lee, andW. Lou. Secure deduplication with efficient and reliable convergent key management. In *IEEE Transactions on Parallel and Distributed Systems*, 2013.
[13] libcurl. http://curl.haxx.se/libcurl/.
[14] C. Ng and P. Lee. Revdedup: A reverse deduplication storage system optimized for reads to latest backups. In *Proc. of APSYS*, Apr 2013.