Analysis of Big Data DE Duplication on Cloud Using the Privacy-Preserving Encrypted Files

Rupam Bhor[1], Dr.Gayatri Bhandari[2]
[1, 2] Dept of Computer Engineering
[1, 2] JSPM's, BSIOTR, Wagholi, Pune

Abstract- *Today is the most important issue in cloud computing is duplication for any organization, so we analysis this issue an avoid the reparative files on cloud storage. Avoidance of the file is advantages the cloud size issue. To protect the confidentiality of sensitive data while supporting deduplication, the convergent encryption technique has been proposed to encrypt the data before outsourcing. To better protect data security, on cloud storage. In this system we check the duplicate file on cloud storage also security apply using encryption. We use the AES encryption algorithm for encrypt the file simultaneously we check the duplicate file using the hashing algorithm. Also enhanced this system using recover option, cloud provide the deleted file backup on requesting. This paper study on the plain text as a input the system for checking the duplicate file, we next stage we analysis the encrypted file as a input to the system and find the duplicate file on the cloud storage.*

Keywords- Duplication, Authorized Duplicate Check, Confidentiality, Cloud computing.

I. INTRODUCTION

The Cloud is network for storage, access resources, where data is stored in pools of storage which are generally hosted by third parties. Cloud storage provides users with benefits, ranging from cost saving and simplified convenience, to mobility opportunities and scalable service. These properties used for customers to use and storage their personal data to the cloud storage: according to the analysis result in worlds, the volume of data in cloud is expected to achieve 40 trillion gigabytes in 2020. Even though cloud storage system has been widely used, it fails to accommodate some main emerging needs such as the abilities of auditing integrity of uploaded data cloud files by cloud clients and detecting duplicated files by cloud servers. We analysis both problems below. The first problem is integrity auditing in the cloud computing. The cloud server is able to remove clients from the heavy burden of storage management and maintenance.

The cloud storage used for storage is that the data is transferred via Internet and stored in an uncertain domain, not under control of the clients at all, which raises clients great concerns on the integrity of their data. These concerns originate from the fact that the cloud storage is susceptible to security threats from both outside and inside of the cloud [1], and some data loss from the clients may be hidden by the uncontrolled cloud servers to maintain the reputation. The most important thing is that for an ordinary clients the data which is rarely accessed is deliberately deleted by the servers to maintain the cost and space Considering the large size of the outsourced data files and the clients' constrained resource capabilities, the first problem is as how can the client efficiently perform periodical in verifications even without the local copy of data files. The second problem is secure deduplication. The increased volumes of data stored at remote cloud servers accompanies the rapid adoption of cloud services is.

According to the last survey of EMC the most of the remotely stored files are deduplicated. [2], Recently the 75% of the digital data is deduplicated. Due to this the term came that is deduplication in which the cloud servers just keep only one file and keeps the link of that file for the user's who wants the same file to store. Due to this it leads to a number of threats affecting the storage system [3][2], for example, a server telling the client that it does not need to send or store the file which is same as other user and it can be dangerous sometimes.. These attacks originate from the proof that client owns a file that totally use static or we can say a hash code. [3]. Thus, the second problem is generalized as how can the cloud servers efficiently confirm that the client owns the uploaded file before creating a link to this file for him/her.

II. REVIEW OF LITERATURE

Our work is related to both integrity auditing and secure deduplication, we review the works in both areas in the following subsections, respectively. 2.1 Intigrity Auditing The definition of provable data possession (PDP) was developed by Ateniese et al. [1][2] for assuring that the cloud servers possess the target files without retrieving or downloading the whole data. Essentially, PDP is a probabilistic proof protocol by sampling a random set of blocks and asking the servers to prove that they exactly possess these blocks, and the verifier

only maintaining a small amount of metadata is able to perform the integrity checking.

After Ateniese et al.'s proposal [1], several works concerned on how to realize PDP on dynamic scenario: Ateniese et al. [2] proposed a dynamic PDP schema but without insertion operation; Erway et al. [3] improved Ateniese et al.'s work [2] and supported insertion by introducing authenticated flip table; A similar work has also been contributed in [4]. Nevertheless, these proposals [1][2][3][4] suffer from the computational overhead for tag creation at the client. To fix this issue, Wang et al. [5] presented proxy PDP in public clouds. Zhu et al. [6] presented the cooperative PDP in multi-cloud storage. Another line of work supporting integrity auditing is proof of retrievability (POR) [7]. Compared with PDP, POR not merely assures the cloud servers possess the target files, but also guarantees their full recovery. In [7], clients apply erasure codes and create authenticators for each block for verifiability and retrievability.

In order to get efficient data dynamics, Wang et al. [8] improved the POR model by manipulating the classic Merkle hash tree construction for block tag authentication. Xu and Chang [9] presented to improve the POR schema in [7] with polynomial commitment for reducing communication cost. Stefanov et al. [10] proposed a POR protocol over authenticated file system subject to frequent changes.

Azraoui et al. [11] combined the privacy-preserving word search algorithm with the insertion in data segments of randomly created short bit sequences, and developed a new POR protocol. Li et al. [12] considered a new cloud storage architecture with two independent cloud servers for integrity auditing to reduce the computation load at client side. Recently, Li et al. [13] used the key-disperse paradigm to fix the issue of a significant number of convergent keys in convergent encryption.

III. SYSTEM ARCHITECTURE / SYSTEM OVERVIEW

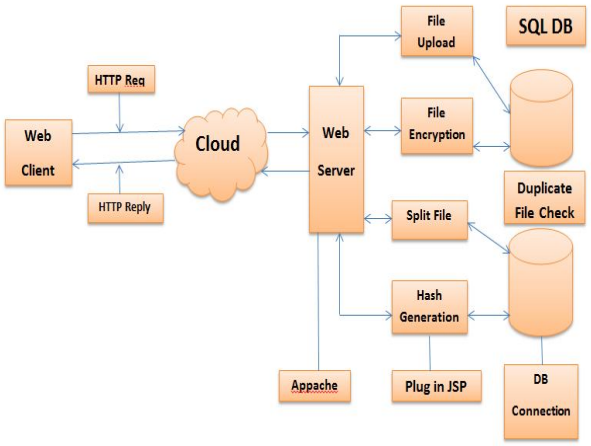

Fig 1. System architecture

1) File Uploading Protocol: This protocol aims at allowing clients to upload files via the auditor. Specifically, the file uploading protocol includes three phases:

I. Phase 1 (cloud client → cloud server): Client takes the duplicate check with the cloud server to confirm if such a file is stored in cloud storage or not before uploading a file. If there is a duplicate, another protocol called Proof of Ownership will be run between the client and the cloud storage server. Otherwise, the following protocols (including phase 2 and phase 3) are run between these two entities.

II. Phase 2 (cloud client → authority): Client uploads files to the auditor, and receives a receipt from authority.

III. Phase 3 (authority → cloud server): Authority helps generate a set of tags for the uploading file, and send them along with this file to cloud server.

IV. MATHEMATICAL MODEL

System Description:

Input:
Upload file ()
U : Upload file on cloud.
E : Encryption File.
S : Splitting file for security.
H : Hash value for each file.

Output:
Check Duplicate file on cloud storage

Input
Function Recovery (id, request, file)
ID : unique id for each file.
Request : User request for recovery of file.
File : Check file on cloud.

Output:
File will recover to data owner.

Encryption Process:

KeyGenCE(M) → K is the key generation algorithm that maps a data copy M to a convergent key K;

EncCE(K,M) → C is the encryption algorithm that takes both the convergent key K and the data copy M as inputs and then outputs cipher text C;

DecCE(K,C) → M is the decryption algorithm that takes both the cipher text C and the convergent key K as inputs and then outputs the original data copy M;

TagGen (M) → T(M) is the tag generation algorithm that maps the original data copy M and outputs a tag T(M)

V. SYSTEM ANALYSIS

We have created system in java. Data is stored in mysql database. We have created a web application with local server. Web application that communicates with local server and Trustee Server using REST API. We have uploaded text document on cloud. We have evaluated time required for tag generation and file deduplication checking for different file sizes. Here we also calculate the file each file size for analysis purpose.

Table 1. Calculate the file size

File Name	File Size In Bytes
Client.txt	3734
Server.txt	3736
csp.txt	3005
module.txt	3007
cloud.txt	3291
service.txt	5398
system.txt	309
Total size	25266

VI. CONCLUSION

Our propose technique provides data security using data encryption in cloud environment. For effective usage of storage space we provide de-duplication check at file level. We also provide new de-duplication constructions supporting authorized duplicate check in cloud architecture, in which the duplicate-check is done at local cloud server. This avoids multiple transaction of file tag over network while checking deduplication. We introduce a relative addressing method in which data will check at entry level when user uploading phases.

VII. ACKNOWLEDGMENT

I wish to express my profound thanks to all who helped us directly or indirectly in making this paper. Finally I wish to thank to all our friends and well-wishers who supported us in completing this paper successfully I am especially grateful to our guide Dr.Gayatri Bhandari Madam for her time to time, very much needed, valuable guidance. Without the full support and cheerful encouragement of my guide, the paper would not have been completed on time.

REFERENCES

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable data possession at untrusted stores,” in Proceedings of the 14th ACM Conference on Computer and Communications Security, ser. CCS ’07. New York, NY, USA: ACM, 2007, pp. 598– 609.

[2] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, “Scalable and efficient provable data possession,” in Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks, ser. SecureComm ’08. New York, NY, USA: ACM, 2008, pp. 9:1– 9:10.

[3] C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, “Dynamic provable data possession,” in Proceedings of the 16th ACM Conference on Computer and Communications Security, ser. CCS ’09. New York, NY, USA: ACM, 2009, pp. 213–222.

[4] F. Seb´e, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, “Efficient remote data possession checking in critical information infrastructures,” IEEE Trans. on Knowl. and Data Eng., vol. 20, no. 8, pp. 1034–1038, 2008.

[5] H. Wang, “Proxy provable data possession in public clouds,” IEEE Transactions on Services Computing, vol. 6, no. 4, pp. 551–559, 2013.

[6] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231– 2244, 2012.

[7] H. Shacham and B. Waters, "Compact proofs of retrievability," in Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, ser. ASIACRYPT '08. Springer Berlin Heidelberg, 2008, pp. 90– 107.

[8] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in Computer Security – ESORICS 2009, M. Backes and P. Ning, Eds., vol. 5789. Springer Berlin Heidelberg, 2009, pp. 355–370.

[9] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ser. ASIACCS '12. New York, NY, USA: ACM, 2012, pp. 79– 80.

[10] E. Stefanov, M. van Dijk, A. Juels, and A. Oprea, "Iris: A scalable cloud file system with efficient integrity checks," in Proceedings of the 28th Annual Computer Security Applications Conference, ser. ACSAC '12. New York, NY, USA: ACM, 2012, pp. 229–238.

[11] M. Azraoui, K. Elkhiyaoui, R. Molva, and M. O¨ nen, "Stealthguard: Proofs of retrievability with hidden watchdogs," in Computer Security - ESORICS 2014, ser. Lecture Notes in Computer Science, M. Kutyłowski and J. Vaidya, Eds., vol. 8712. Springer International Publishing, 2014, pp. 239–256.

[12] J. Li, X. Tan, X. Chen, and D. Wong, "An efficient proof of retrievability with public auditing in cloud computing," in 5th International Conference on Intelligent Networking and Collaborative Systems (INCoS), 2013, pp. 93–98. International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 02 Issue: 09 | Dec-2015 www.irjet.net p-ISSN: 2395-0072 © 2015, IRJET ISO 9001:2008 Certified Journal Page 654.

[13] J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 6, pp. 1615–1625, June 2014.