IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 11, November 2015*

# Deduplication Techniques in Storage System

**Deepali Choudhari[1], R. W. Deshpande[2]**

M.E. Student, Dept. of IT, Sidhhant College of Engineering, Pune, India[1]

Professor, Dept. of IT, Sidhhant College of Engineering, Pune, India[2]

**Abstract:** Ever increasing volume of back up data in cloud storage may be a vital challenge. There is a need of data management as back up windows are shrinking due to growth of information. So, to make data management scalable deduplication concept is used. It is a technique of keeping only one unique instance of data copy by detecting identical data copies and eliminating those so that it could improve storage utilization, system performance of storage system. There are different schemes introduced by people. This paper surveys these different deduplication approaches.

**Keywords:** deduplication, convergent encryption, cloud storage, cryptographic.

## I. INTRODUCTION

Cloud computing gives boundless virtualized plan of action to client as administrations over the entire web while concealing the stage and executing subtle elements. Today by increasing the volume of information in cloud storage created problem for duplicate data as data is collected from heterogeneous sources. Cloud storage is getting popular more and more as it is low cost and on-demand use of large storage. To make data management scalable in cloud computing, the deduplication concept is used. According to the analysis report of IDC, the volume of data in the world is expected to reach 40 trillion gigabytes in 2020 [1]. Today's commercial cloud storage services, such as Dropbox, Google Drive and Mozy, have been applying deduplication to save the network bandwidth and the storage cost with client-side deduplication. Deduplication means duplicate data is eliminated, a pointer is created to reference a data that is backed up. In order to have a secure storage of deduplicated data over a cloud computing we use the encryption/decryption technique.

Deduplication can take place at 1) File level, in this it detects redundant data within the files or 2) Block level, in this it detects redundant data across the files and removes those data of identical data files. In file level deduplication, system first check within the files in storage for duplicates and if there is no duplicates in file then block level deduplication is done. It operates on basis of sub-file level in which check duplicates or identical data across the files. File is being broken into blocks or chunks or segments which are then compared to previously stored data.

Before upload the data to the server user need to encrypt the data and then upload their data on to the cloud. Encryption is the process of converting a plaintext into a ciphertext [2]. Traditional encryption is contradictory to deduplication as it requires different users to encrypt their data with their own keys.Convergent encryption is mainly used for the deduplication process[3]. To keep the confidentiality of sensitive data while supporting the deduplication, to encrypt the databefore outsourcing convergent encryption technique has been proposed. Convergent encryption is well known concept. It checks the duplicate by tag value of data file produced by users. A Proof of ownership protocol is executed to identify data copy owner that is which user owns the data file [7].A number of deduplication systems have been proposed based on various deduplication strategies are explained in literature survey.

## II. METHODS USED IN DEDUPLICATION

Following are the some basic methods used in deduplication:

### A. Symetric Encryption

Symmetric encryption uses a common secret key k to encrypt and decrypt information. A symmetric encryption scheme made up of three primary functions.

1) KeyGen SE ($1\lambda$)→ : k is the key generation algorithm that generates k using security parameter $1\lambda$;
2) Enc SE (k, M)→ C: is the symmetric encryption algorithm that takes the secret k, and message M and then outputs the ciphertext C, and
3) Dec SE (k, C) → M: is the symmetric decryption algorithm that takes the secret k and ciphertext C and then outputs the original message M.

Each user encrypts the data with their own encryption algorithm. In these identical data copies that produce the different ciphertext, this makes the deduplication process impossible.

### B. Convergent Encryption

Convergent encrption encrypts a data copy with a convergent key, which is derived by cryptographic hash value of the content of the data itself [3].In addition user derives a tag for the data copy, such that tag will be used to detect duplicates. After key generation and data encryption, users retain the keys and send the tag to the server side to check the identical copy. It is assumed that if two copies are identical then their corresponding tag values are also identical. Since encryption is deterministic, identical data copies will generate the same convergent key and same cipher text. This allows the cloud to perform deduplication on cipher texts can only be decrypted by corresponding data owners with theirs convergent keys.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 11, November 2015*

C. Proof of ownership

Proof of ownership allows ownership of data copies to the server side. When tag value is same in storage then it should be proven that which user, owner owns that file.

## III.LITERATURE SURVEY

In [3], convergent encryption is used via A cryptographic primitive, Message-Locked Encryption(MLE), is a symmetric encryption scheme where the key under which encryption and decoding are performed is itself drawn from the message.MLE scheme is depicted in fig1. A MLE scheme MLE = (P,K,E,D, T ) is a five-tuple algorithm. On input 1 the parameter generation algorithm P returns a public parameter P. On input P and a message M, the key-generation algorithm K returns a message-derived key K. This key is stored by user securely. Then on inputs P, K, M the encryption algorithm E returns a ciphertext C. This encrypted data send to server. On inputs P, K and a ciphertext C, the decryption algorithm D returns original message M. Then tag is calculated by the tag generation algorithm and sever provides that pointer or tag. In this way data file is uploaded. While at the time of download client receives file using that pointer. In light of this establishment, we make both common sense and hypothetical commitments. On the pragmatic side, we give ROM security investigations of a characteristic group of MLE plans that incorporates conveyed plans. On the hypothetical side the test is standard model.

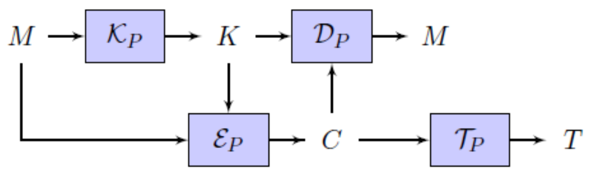
$$\begin{tikzcd}
M \arrow[r] \arrow[dr] & \mathcal{K}_P \arrow[r] & K \arrow[r] \arrow[d] & \mathcal{D}_P \arrow[r] & M \\
 & & \mathcal{E}_P \arrow[r] & C \arrow[u] \arrow[r] & \mathcal{T}_P \arrow[r] & T
\end{tikzcd}$$

Fig. 1 MLE scheme

To protect data confidentiality, [4] worked on it by transforming the predicatable message into unpredicatable message. In their system, another third party called the key server was introduced to generate the file tag for the duplicate check. It provides secure deduplicated storage resisting brute-force attacks, and realize it in a system called DupLESS(Duplicateless Encryption for simple storage). In DupLESS, clients encrypt under message based keys obtained from a key-server instead of setting keys to be hashes of messages. Clients themselves communicate to KS. DupLESS employs an oblivious PRF (OPRF) protocol between the KS and clients, which ensures that the KS learns nothing about the client inputs or the resulting PRF outputs, and that clients learn nothing about the key. It enables clients to store encrypted data with an existing service, have the service perform deduplication on their behalf and yet achieves strong confidentiality guarantees. Client store data file by RSA OPRF protocol with the KS to compute a message derived key, then encrypt with key to produce a ciphertext. The client's secret key will be used to encrypt key to produce a key encapsulation ciphertext. Both key encapsulation ciphertext and encrypted data file are stored on the SS. It works on two KS protocols that clients can use while encrypting files. HTTP based RESTful web interface and custom protocol built over UDP.

For the proof of ownership [7] enables users to prove their ownership of data copies to the storage server. Specifically, Proof of ownership is implemented as an interactive algorithm run by a user and a storage server. To solve the problem of using a small hash value as a proxy for the entire file, they designed a solution where a client proves to the server that it indeed has the file. We call a proof mechanism that prevents such leakage amplification a proof of ownership (PoW). We note that this is somewhat similar to proofs of retrievability (PORs) [9,10] and proofs of data possession (PDPs) [11] with a role reversal (the client is the prover rather than the server). However, this role reversal between client and server is significant. In this work a client can prove to a server that it has a copy of a file without actually sending the file. This can be used to counter attacks on file-deduplication systems where the attacker obtains a “short summary” of the file and uses it to fool the server into thinking that the attacker owns the entire file. It focused on different types of attacks.It does not consider any leakage setting.

In a proof of ownership scheme, any owner of the same file F can prove to the cloud storage that he/she owns file F in a robustand efficient way, in the bounded leakage setting wherea certain amount of efficiently-extractable information aboutfile F is leaked. It adopted in [8] a weaker leakage setting, They allow a bounded amount one-time leakage of a target file before this system starts to execute. An important security concern in cross-user client-side deduplication (confidentiality of users' sensitive files against both outside adversaries and the honest-but-curious cloud storage server in the bounded leakage model) of encrypted files in the cloud storage is addressed.

A novel encryptionscheme that provided in [5] that support differential security for popular and unpopular data. For popular data that are not particularly sensitive, the traditional conventional encryption is performed. A careful consideration of potential attacks such as Data Leakage and poison attacks are needed that target privacy preservation and data confidentiality disclosure. Proof of Ownership (PoW), based on the joint use of convergent encryption and the Merkle-based Tree, for improving data security in cloud storage systems, providing dynamic sharing between users and ensuring efficient data deduplication. Fig 2 shows Merkle Hash Tree is illustrated. It provides 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.
It consists in using the Merkle-based Tree over encrypted

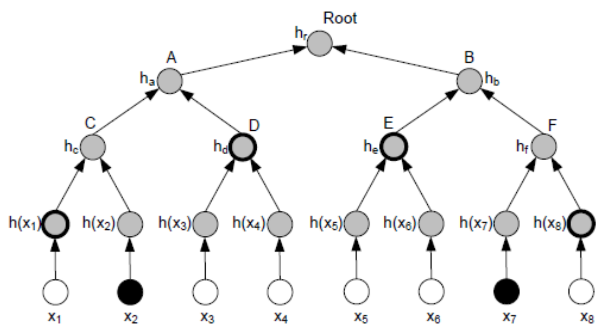

Fig. 2 Merkle-based tree the leaf nodes h(x1), . . .. h(xn) as the left-to-right sequence.

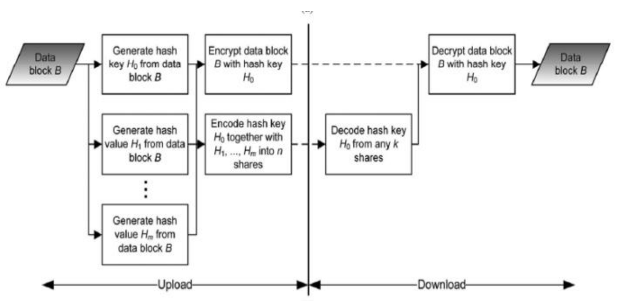
Data block B
Generate hash key $H_0$ from data block B
Encrypt data block B with hash key $H_0$
Generate hash value $H_1$ from data block B
Encode hash key $H_0$ together with $H_1, \dots, H_m$ into n shares
Generate hash value $H_m$ from data block B
Decode hash key $H_0$ from any k shares
Decrypt data block B with hash key $H_0$
Data block B
Upload
Download

Fig. 3 Data Flow Diagram of Dekey Construction

data, in order to derive a unique identifier of outsourced data. It supports data deduplication, as it employs preverification of data existence, in cloud servers, which is useful for saving bandwidth. 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. They proposed and implemented, on OpenStack Swift, a new client-side deduplication scheme for securely storing and sharing outsourced data via the public cloud. They built a simulated cloud storage framework, based on OpenStack Storage system (Swift). It is a cloud based storage system, which stores data and allows write, read, and delete operations on them.

The reliability in deduplication has been addressed in [6] by Jin Ji et al. It shows how to achieve reliable key management in deduplication. However, they did not mention about the application of reliable deduplication for encrypted files. It addresses the key-management issue in block-level deduplication by distributing these keys across multiple servers after encrypting the files. Convergent encryption provides a viable option to enforce confidentiality while realizing deduplication. It encrypts/decrypts a data copy with a convergent key, which is derived by cryptographic hash value of the content of the data itself. After key generation and data encryption, users retain the keys and send the cipher text to the cloud. Since encryption is deterministic, identical data copies will generate the same convergent key and same cipher text. This allows the cloud to perform deduplication on cipher texts can only be decrypted by corresponding data owners with theirs convergent keys.

In [6] Dekey is designed to efficiently and reliably maintain convergent keys. It is a new construction in which users do not need to manage any keys on their own but instead securely distribute the convergent keys shares across multiple key management severs that is KM-CSPs. If multiple users share the same block, they can access the same corresponding convergent key. This significantly reduces the overhead for convergent keys. It allows convergent keys to remain accessible even if any subset of KM-CSPs fails.

Generally at the time of storing the data file on storage system, after encryption of data block with hash key each hash key $H_0$ is encrypted with user's master key while in Dekey approach n shares that is $H_1$, $H_2 \dots H_m$ of $H_0$ are generated after encryption of data block and hash key.

Then this n shares together encode with hash key $H_0$. When user want that data he just simply decode hash key first from any k shares. Using this decoded hash key data block is recovered.

The secret sharing technique is utilized To protect data confidentiality [12]. A file is divided into pieces say n by using the technique of secret sharing in which data can be reconstructable using some pieces say k only but can not be reconstruct by k – 1 pieces. Such scheme is called as (k, n) threshold scheme. This scheme is based on polynomial interpolation.

Paper [13] addresses the problems of identifying and coalescing identical files in the Farsite distributed filesystem, for the purpose of reclaiming storage space consumed by incidentally redundant content. Farsite is a secure, scalable, serverless file system that logically functions as a centralized file server but that is physically distributed among a networked collection of desktop workstation. They includes convergent encryption, which enables duplicate files to coalesced into the space of a single file, even if the files are encrypted with different users' keys, and SALAD, a Self- Arranging, Lossy, Associative Database for aggregating file content and location information in a decentralized, scalable, fault-tolerant manner. It shows results from large-scale simulation experiments using file content data collected from a set of 585 desktop file systems.

The traditional deduplication methods cannot be directly extended and applied in distributed and multi-server systems. If it is used, it cannot resist the collusion attack launched by multiple servers. In [14] a file is first split and encoded into fragments by using the technique of Ramp secret sharing, instead of encryption mechanisms. These shares are then distributed across multiple independent storage servers. Tag, a short cryptographic hash value of the content will also be computed and sent to each storage server as the fingerprint of the fragment stored at each server. Only the data owner who first uploads the data is required to compute and distribute such secret shares, while all following users who own the same data copy do not need to compute and store these shares any more. To recover data copies, users must access a minimum number of storage servers through authentication and obtain the secret shares to reconstruct the data.

## IV. CONCLUSION

In computing, data deduplication is a specialized data compression technique for eliminating duplicate copies of repeating data. Related and somewhat synonymous terms are intelligent (data) compression and single-instance (data) storage. This 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. In the deduplication process, unique chunks of data, or byte patterns, are identified and stored during a process of analysis. In this way, in this paper different approaches towards data uploading on cloud storage system are studied.

## ACKNOWLEDGMENT

I submit my gratitude and sincere thanks to my guide for her constant motivation and support. I truly appreciate and value her esteemed guidance and encouragement. I would like to thanks to my staff members also for their support and finally to my family and friends.

## REFERENCES

[1] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digi tal shadows, and biggest growth in the far east," http://www.emc.com/collateral/analyst-reports/idcthe-digital-universe-in-2020.pdf, Dec 2012.
[2] S. Kamara and K. Lauter, ''Cryptographic Cloud Storage,'' in Proc. Financial Cryptography: Workshop Real-Life Cryptograph. Protocols Standardization, 2010, pp. 136-149.
[3] "Message-locked encryption and secure deduplication," in EUROCRYPT, 2013, pp. 296–312.
[4] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Dupless: Server-aided encryption for deduplicated storage," in USENIX Security Symposium, 2013.
[5] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in Technical Report, 2013.
[6] J. Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," in IEEE Transactions on Parallel and istributed Systems, 2014, pp. vol. 25(6), pp. 1615–1625.
[7] 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.
[8] J. Xu, E.-C. Chang, and J. Zhou, "Weak leakage-resilient client-side deduplication of encrypted data in cloud storage," in ASIACCS, 2013, pp. 195–206.
[9] A. Juels and B. S. Kaliski, Jr. Pors: proofs of retrievability for large files. In ACM CCS '07, pages 584–597. ACM, 2007.
[10] H. Shacham and B. Waters. Compact proofs of retrievability. In ASIACRYPT '08, pages 90–107. Springer-Verlag, 2008.
[11] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song. Provable data possession at untrusted stores. In ACM CCS '07, pages 598–609. ACM, 2007.
[12] A. Shamir, "How to share a secret," Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979.
[13] 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, 2002, pp. 617–624.
[14] Jin Li, Xiaofeng Chen, Xinyi Huang, Shaohua Tang and Yang Xiang, "Secure Distributed System with Improved Reliability" in IEEE Transaction on Computers Volume: PP Year 2015.