International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 20 Issue 2 – FEBRUARY 2016.

EFFICIENT DISTRIBUTED DEDUPLICATION SYSTEM WITH HIGHER RELIABILITY MECHANISMS IN CLOUD

P.Nivetha

M.Tech Information Technology, Vivekananda College of Engineering for Women, Tiruchengode-637205

***Abstract*--Data deduplication is a methodology of reducing storage needs by eliminating redundant data. Only one unique instance of the data should be retained on storage media, such as disk or tape. Redundant data is replaced with a pointer to the data copy and it has been widely used in many cloud storage technique to reduce the amount of storage space and save bandwidth. To maintain the confidentiality of sensitive data while supporting deduplication, the convergent encryption technique has been proposed to encrypt data while outsourcing. To better protect data security, this work makes an attempt to address the problem of authorized data deduplication. The several deduplication techniques are implemented in Hybrid Cloud architecture. It uses hashing technique to maintain the uniqueness of textual data and Transformation techniques to maintain the same in images.**

Index Terms – Deduplication, Reliability.

I. INTRODUCTION

Cloud Computing is a composition of two or more agents like clouds (private, community or public), users and auditors that remain distinct but are bound together and offering the benefits of multiple deployment agents. Hybrid cloud can also be defined as ability to connect collocation, managed and dedicated service with cloud resources attached with it . This hybrid cloud services crosses isolation and provider boundaries so that it can't be simply gathered in one category of private, public, or community cloud service. It allows extending either the capability or the capacity of a cloud service, by aggregation, assimilation or customization with a different cloud service.

Varieties of use cases for hybrid cloud composition exist. For example, an organization may lay up susceptible client data in the warehouse on a private cloud application, but interconnects that application to business intelligence applications which is provided on a public cloud as a software service. This example of hybrid cloud storage extends the capability of the enterprise to deliver specific business services through the addition of externally available public cloud services.

Yet another example of hybrid cloud computing is about IT organizations which uses public cloud computing resource to meet capacity requirements that cannot be met by private cloud. This capabilities enables hybrid clouds enables employing of cloud bursting for scaling across number of clouds. Cloud bursting is an application deployment model which runs in a private cloud or data center along with "bursts" to a public cloud when there is a claim for computing capacity increases. The primary advantages of cloud bursting and a hybrid cloud model is that the association can pay for work out possessions only when they are needed.

To make efficient data management in cloud computing, deduplication of data has been a well-known technique and has attracted more attentions recently. Data deduplication is specialized data compression based technique used for eliminating duplicate copies of repeating data in storage. The scheme is to expand storage utilizations and can be also applied to network data transfer to diminish the number of bytes that should be sent. Instead of maintaining multiple data copy with the same content, deduplication remove unneeded data by keeping only one physical copy and to refer other redundant data to that copies. Deduplication occurs at each of the file level and the block level. For file level, it eliminates duplicate copies of same file. Deduplication can take place at block level too, which eliminate duplicate block of data that occur in non-identical filesystem.

Even though data deduplication brings number of benefits, security with privacy concerns arise as user's sensitive data are vulnerable to both inside and outside attacks. Traditional encryptions, while providing confidentiality of data, is incompatible with data deduplication system. Traditional encryption require different users to encrypt their datum with their own keysets. Thus, identical data copy for different users will lead to different cipher text, making deduplication daunting. Convergent encryption has been used to implement data privacy while making deduplication technique feasible. It encrypts/ decrypts a data copy with a convergent keys, which is obtained by computing cryptographic hash value of the content of data copy. After key creation and data encryption, users retain the keys and send the ciphertexts to the cloud. This work perform duplication elimination on both image and text data so that the user cannot store this type of data in a repeated way which increase the overhead of the server.

II. THE DISTRIBUTED DEDUPLICATION SYSTEMS-THE EXISTING SYSTEM

2.1 Secret Sharing Scheme

There are two algorithms in a secret sharing scheme, which are Share and Recover. The secret is divided and shared by using Share. With enough shares, the secret can be extracted and recovered with the algorithm of Recover. In our implementation, we will use the Ramp secret sharing scheme (RSSS) to secretly split a secret into shards. Specifically, the (n, k, r)-RSSS (where $n > k > r \geq 0$) generates n shares from a secret so that (i) the secret can be recovered from any k or more shares, and (ii) no information about the secret can be deduced from any r or less shares. Two algorithms, Share and Recover, are defined in the (n, k, r)-RSSS.

- Share divides a secret S into $(k - r)$ pieces of equal size, generates r random pieces of the same size, and encodes the k pieces using a non-systematic k-of-n erasure code into n shares of the same size;
- Recover takes any k out of n shares as inputs and then outputs the original secret S.

It is known that when $r = 0$, the $(n, k, 0)$-RSSS becomes the (n, k) Rabin's Information Dispersal Algorithm (IDA) [9]. When $r = k - 1$, the $(n, k, k - 1)$-RSSS becomes the (n,k) Shamir's Secret Sharing Scheme (SSSS) [11].

2.2 Tag Generation Algorithm

In our system below, two kinds of tag generation algorithms are defined, that is, TagGen and TagGen'. TagGen is the tag generation algorithm that maps the original data copy F and outputs a tag $T(F)$. This tag will be generated by the user and applied to perform the duplicate check with the server. Another tag generation algorithm TagGen' takes as input a file F and an index j and outputs a tag. This tag, generated by users, is used for the proof of ownership for F.

2.3 The File-level Distributed Deduplication System

To support efficient duplicate check, tags for each file will be computed and are sent to S-CSPs. To prevent a collusion attack launched by the S-CSPs, the tags stored at different storage servers are computationally independent and different. We now elaborate on the details of the construction as follows.

System setup In our construction, the number of storage servers S-CSPs is assumed to be n with identities denoted by $id_1, id_2, \cdots, id_n$, respectively. Define the security parameter as 1 and initialize a secret sharing scheme SS = (Share, Recover), and a tag generation algorithm TagGen. The file storage system for the storage server is set to be $\perp$.

File Upload To upload a file F, the user interacts with S-CSPs to perform the deduplication. More precisely, the user firstly computes and sends the file tag $\phi_F = \text{TagGen}(F)$ to S-CSPs for the file duplicate check.

- If a duplicate is found, the user computes and sends $\phi_{F;id_j} = \text{TagGen}'(F, id_j)$ to the j-th server with identity id_j via the secure channel for $1 \leq j \leq n$ (which could be implemented by a cryptographic hash function $H_j(F)$ related with index j). The reason for introducing an index j is to prevent the server from getting the shares of other S-CSPs for the same file or block, which will be explained in detail in the security analysis. If $\phi_{F;id_j}$ matches the metadata stored with ϕ_F, the user will be provided a pointer for the shard stored at server id_j.

- Otherwise, if no duplicate is found, the user will proceed as follows. He runs the secret sharing algo-rithm SS over F to get $\{c_j\} = \text{Share}(F)$, where c_j is the j-th shard of F. He also computes $\phi_{F;id_j} = \text{TagGen}'(F, id_j)$, which serves as the tag for the j-th S-CSP. Finally, the user uploads the set of values $\{\phi_F, c_j, \phi_{F;id_j}\}$ to the S-CSP with identity id_j via a secure channel. The S-CSP stores these values and returns a pointer back to the user for local storage.

File Download. To download a file F, the user first downloads the secret shares $\{c_j\}$ of the file from k out of n storage servers. Specifically, the user sends the pointer of F to k out of n S-CSPs. After gathering enough shares, the user reconstructs file F by using the algorithm of Recover($\{c_j\}$).

This approach provides fault tolerance and allows the user to remain accessible even if *any* limited subsets of storage servers fail.

2.4 The Block-level Distributed Deduplication System

In this section, we show how to achieve the fine-grained block-level distributed deduplication. In a block-level deduplication system, the user also needs to firstly perform the file-level deduplication before uploading his file. If no duplicate is found, the user divides this file into blocks and performs block-level deduplication. The system setup is the same as the file-level deduplication system, except the block size parameter will be defined additionally. Next, we give the details of the algorithms of File Upload and File Download.

File Upload. To upload a file F, the user first performs the file-level deduplication by sending ϕ_F to the storage servers. If a duplicate is found, the user will perform the file-level deduplication, such as that in Section 3.2. Otherwise, if no duplicate is found, the user performs the block-level deduplication as follows.

He firstly divides F into a set of fragments $\{B_i\}$ (where $i = 1, 2, \cdots$). For each fragment B_i, the user will perform a block-level duplicate check by comput-ing $\phi_{Bi} = \text{TagGen}(B_i)$, where the data processing and duplicate check of block-level deduplication is the same as that of file-level deduplication if the file F is replaced with block B_i.

Upon receiving block tags $\{\phi_{Bi}\}$, the server with identity id_j computes a block signal vector σ_{Bi} for each i.

- i) If $\sigma_{Bi} = 1$, the user further computes and sends $\phi_{Bi;j} = \text{TagGen}'(B_i, j)$ to the S-CSP with identity id_j. If it also matches the corresponding tag stored, S-CSP returns a

block pointer of B_i to the user. Then, the user keeps the block pointer of B_i and does not need to upload B_i.

- ii) If σ_{Bi} =0, the user runs the secret sharing al-gorithm SS over B_i and gets $\{c_{ij}\}$ = Share(B_i), where c_{ij} is the j-th secret share of B_i. The user also computes $\phi_{Bi;j}$ for $1 \leq j \leq n$ and uploads the set of values $\{\phi_F , \phi_{F;idj} , c_{ij} , \phi_{Bi;j}\}$ to the server id_j via a secure channel. The S-CSP returns the corresponding pointers back to the user.

File Download. To download a file $F = \{B_i\}$, the user first downloads the secret shares $\{c_{ij}\}$ of all the blocks B_i in F from k out of n S-CSPs. Specifically, the user sends all the pointers for B_i to k out of n servers. After gathering all the shares, the user reconstructs all the fragments B_i using the algorithm of Recover($\{\cdot\}$) and gets the file $F = \{B_i\}$.

III. THE PROPOSED SYSTEM

In this work we have implement SHA(Secured Hash Algorithm) Algorithm to determine the text level duplication by generating a single hash value for the whole file which reduce the overhead of maintaining more number of hash value for a single file. Also, KDC(Key Distribution centre) is used to generate a random number using registration details of the user which will be used to maintain the authentication of the user. (SWQ)Scalar Wavelet Quantization Algorithm is used to avoid the duplication of image. It can be done by converting the image into coefficients so that if the coefficient values are same, then we can judge then we can judge that the images are identical.

3.1 IMPLEMENTATIONS:

The following entities will be involved in this deduplication system:

- Private cloud
- Public cloud
- Data user
- Deduplication system

3.2 Private cloud:

Compared with the traditional deduplication architecture in cloud computing, this is a new entity introduced for facilitating user's secure usage of cloud service. particularly, since the computing resources at data user/owner side are limited and the public cloud is not fully trusted in practice, private cloud is able to provide data user/owner with an execution environment and infrastructure working as an interface between user and the public cloud. The private keys for the privileges are managed by the private cloud, who answers the file token requests from the users. The interface offered by the private cloud allows user to submit files and queries to be securely stored and computed respectively.

3.3 Public cloud:

This is an entity that provides a data storage service in public cloud. The public cloud provides the data outsourcing service and stores data on behalf of the users. To reduce the storage cost, the public cloud eliminates the storage of redundant data via deduplication and keeps only unique data. In this paper, we assume that public cloud is always online and has abundant storage capacity and computation power.

3.4 Data user:

A user is an entity that wants to outsource data storage to the public cloud and access the data later. In a storage system supporting deduplication, the user only uploads unique data but does not upload any duplicate data to save the upload bandwidth, which may be owned by the same user or different users. In the authorized deduplication system, each user is issued a set of privileges in the setup of the system. Each file is protected with the convergent encryption key and privilege keys to realize the authorized deduplication with differential privileges. For more security we use an cryptographic technique called encryption using Jenkins hah function algorithm.

3.5 Deduplication System:

In this deduplication module, both image and text data will be processed to avoid duplication of content. In the image data duplication, the SWQ algorithm is used to determine the coefficient of the images. Once the coefficients are determined they will be compared and the similarity ratio will conclude whether to consider the second image or to discard.

Also text based deduplication is performed by generating the hash value using SHA algorithm for each document. If the hash values for both the documents are same, then it will be discarded otherwise it will be considered for storage.

Deduplication According to the data granularity, deduplication strategies can be categorized into two main categories: file-level deduplication and block-level deduplication, which is nowadays the most common strategy. In block-based deduplication, the block size can either be fixed or variable. Another categorization criteria is the location at which deduplication is performed: if data are deduplicated at the client, then it is called source-based deduplication, otherwise target-based. In source-based deduplication, the client first hashes each data segment he wishes to upload and sends these results to the storage provider to check whether such data are already stored: thus only "undeduplicated" data segments will be actually uploaded by the user. While deduplication at the client side can achieve bandwidth savings, it unfortunately can make the system vulnerable to side-channel attacks whereby attackers can immediately discover whether a certain data is stored or not. On the other hand, by deduplicating data at the storage provider, the system is protected against side-channel attacks but such solution does not decrease the communication overhead.

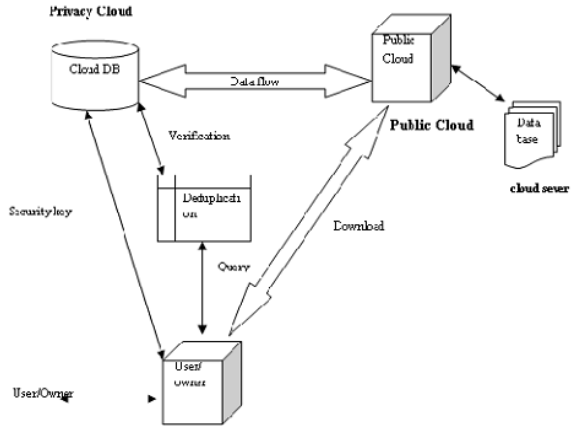
3.5 Secure Hash Algorithm(SHA)

Using Secure Hash algorithm for authentication purpose, SHA is the one of several cryptographic hash functions, most often used to verify that a file has been unaltered. SHA-3 uses the sponge construction,[10][11] in which message blocks are XORed into a subset of the state, which is then transformed as a whole. In the version used in SHA-3, the state consists of a 5×5 array of 64-bit words, 1600 bits total. The authors claim 12.5 cycles per byte on an Intel Core 2 CPU. However, in hardware implementations, it is notably faster than all other finalists.

Keccak's authors have proposed additional, not-yet-standardized uses for the function, including an authenticated encryption system and a "tree" hash for faster hashing on certain architectures. Keccak is also defined for smaller power-of-2 word sizes w down to 1 bit (25 bits total state). Small state sizes can be used to test cryptanalytic attacks, and intermediate state sizes (from w = 8, 200 bits, to w = 32, 800 bits) can be used in practical, lightweight applications.

3.6 Scalar Wavelet Quantization Algorithm (SWQ)

In quantization and entropy coding, the same choices are available to wavelet compression as to other transform-based compression techniques.For quantization, one can choose between scalar and vector quantizers. Similarly, for entropy coding, a variety of lossless bit-packing techniques can be used, such as run-length encoding. In the transform stage, however, wavelet compression differs significantly from other transform based compression techniques. and there is no central authority. Furthermore, any access structure can be expressed in our scheme using the access tree technique. Finally, our scheme relies on the standard complexity assumption, rather than the non-standard complexity assumptions.



IV. CONCLUSION

In this work, 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.We showed that our authorized duplicate check scheme incurs minimal overhead compared to convergent encryption and network transfer**.**

Specifically, in standard transform-based compression such as DCT, the transform is a matrix-vector multiplication, where the vector represents the signal to be transformed and compressed, while the matrix is fixed for any given signal size.

In wavelet compression, on the other hand, the transform is a pair of filters, where one is low-pass and the other is high-pass, and the output of each filter is down sampled by two. Low pass filter is filtering only the low frequency data, high pass extracts only high frequency. Scalar quantization is the dividing signals into individual signal, vector is grouping of signal.

Each of those two output signals can be further transformed similarly, and this process can be repeated recursively several times, resulting in a tree-like structure, called the decomposition tree. Unlike in DCT compression where the transform matrix is fixed, in wavelet transform the designer has the choice of what filter-pair to use and what decomposition tree structure to follow.

3.7 Key Distribution Center

We proposed a privacy-preserving decentralized Key Distribution Center(KDC) scheme to protect the user's privacy. In our scheme, all the user's secret keys are tied to his identifier to resist the collusion attacks while the multiple authorities cannot know anything about the user's identifier. Notably, each authority can join or leave the system freely without the need of reinitializing the system

V. ACKNOWLEDGMENTS

Our thanks to the almighty god, our experts and friends who have contributed towards development of the paper and for their innovative ideas.

REFERENCES

[1] Amazon, "Case Studies," https://aws.amazon.com/solutions/casestudies/#backup.

[2] J. Gantz and D. Reinsel, "The digital universe in 2020: Bigdata, bigger digi tal shadows, and biggest growth in thefar east," http://www.emc.com/collateral/analyst-reports/idcthe-digital-universe-in-2020.pdf, Dec 2012.

[3] M. O. Rabin, "Fingerprinting by random polynomials," Center for Research in Computing Technology, Harvard University, Tech.Rep. Tech. Report TR-CSE-03-01, 1981.

[4] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer,"Reclaiming space fromduplicate files in a serverless distributed file system." in *ICDCS*, 2002, pp. 617–624.

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

[6] ——, "Message-locked encryption and secure deduplication," in *EUROCRYPT*, 2013, pp. 296–312.

[7] G. R. Blakley and C. Meadows, "Security of ramp schemes," in *Advances in Cryptology: Proceedings of CRYPTO '84*, ser. Lecture Notes in Computer Science, G. R. Blakley and D.Chaum, Eds.Springer-Verlag Berlin/Heidelberg, 1985, vol. 196, pp. 242–268.

[8] A. D. Santis and B. Masucci, "Multiple ramp schemes," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1720–1728,Jul. 1999.

[9] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *Journal of the ACM*, vol. 36, no. 2,pp. 335–348, Apr. 1989.

[10] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11,pp. 612–613, 1979.

[11] 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 Distributed Systems*, 2014, pp. vol. 25(6), pp. 1615–1625.

[12] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems." in *ACM Conference on Computer and Communications Security*, Y. Chen, G.Danezis, and V. Shmatikov, Eds. ACM, 2011, pp. 491–500.

[13] J. S. Plank, S. Simmerman, and C. D. Schuman, "Jerasure: A library in C/C++ facilitating erasure coding for storage applications- Version 1.2," University of Tennessee, Tech. Rep. CS-08-627,August 2008.

[14] J. S. Plank and L. Xu, "Optimizing Cauchy Reed-solomon Codes for fault-tolerant network storage applications," in *NCA-06: 5th IEEE International Symposium on Network Computing Applications*, Cambridge, MA, July 2006.

[15] C. Liu, Y. Gu, L. Sun, B. Yan, and D. Wang, "R-admad: High reliability provision for large-scale de-duplication archival storage systems," in *Proceedings of the 23rd international conference on Supercomputing*, pp. 370–379.

[16] M. Li, C. Qin, P. P. C. Lee, and J. Li, "Convergent dispersal:Toward storage-efficient security in a cloud-of-clouds," in *The 6th USENIX Workshop on Hot Topics in Storage and File Systems*, 2014.

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

[18] Z. Wilcox-O'Hearn and B. Warner, "Tahoe: the least-authority filesystem," in *Proc. of ACM StorageSS*, 2008.

[19] A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, and J. C. S.Lui, "A secure cloud backup system with assured deletion and version control," in *3rd International Workshop on Security in Cloud Computing*, 2011.

[20] M. W. Storer, K. Greenan, D. D. E. Long, and E. L. Miller, "Secure data deduplication," in *Proc. of StorageSS*, 2008.

[21] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in *Technical Report*,2013.

[22] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage." *IEEE Security &Privacy*, vol. 8, no. 6, pp. 40–47, 2010.

[23] R. D. Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication." in *ACM Symposium on Information, Computer and Communications Security*, H. Y. Youm and Y. Won, Eds. ACM, 2012, pp. 81–82.

[24] 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.

[25] W. K. Ng, Y. Wen, and H. Zhu, "Private data deduplication protocols in cloud storage." in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, S. Ossowski and P.Lecca, Eds.ACM, 2012, pp. 441–446.

[26] 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. [Online]. Available:http://doi.acm.org/10.1145/1315245.1315318, IEEE Transactions on Computers Volume: PP Year: 2015

[27] A. Juels and B. S. Kaliski, Jr., "Pors: proofs of retrievability for large files," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 584–597. [Online]. Available: http://doi.acm.org/10.1145/1315245.1315317

[28] H. Shacham and B. Waters, "Compact proofs of retrievability," in *ASIACRYPT*, 2008, pp. 90–107.