ISSN: 2321-7782 (Online)
Volume 3, Issue 12, December 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Distributed Deduplication System in Cloudstorage Using Efficient and Reliable Convergent Key Management

Dr. K. Swathi[1]
Professor and Principal,
Department of Computer science &Engineering
Cauvery College of Engineering & Technology, Trichy (TN)

P. Vijayalakshmi[2]
Scholar,
Department of Computer science &Engineering
Cauvery College of Engineering & Technology, Trichy (TN)

Abstract: Data deduplication is a technique for eliminating duplicate copies of data, and has been widely used in cloud storage to reduce storage space and upload bandwidth. However, there is only one copy for each file stored in cloud even if such a file is owned by a huge number of users. As a result, deduplication system improves storage utilization while reducing reliability. Furthermore, the challenge of privacy for sensitive data also arises when they are outsourced by users to cloud. Aiming to address the above security challenges, this paper makes the first attempt to formalize the notion of distributed reliable deduplication system. We propose new distributed deduplication systems with higher reliability in which the data chunks are distributed across multiple cloud servers.The security requirements of data confidentiality and tag consistency are also achieved by introducing a deterministic secret sharing scheme in distributed storage systems, instead of using convergent encryption as in previous deduplication systems. Security analysis demonstrates that our deduplication systems are secure in terms of the definitions specified in the proposed security model. As a proof of concept, we implement the proposed systems and demonstrate that the incurred overhead is very limited in realistic environments.

I. INTRODUCTION

With the explosive growth of digital data, de duplication techniques are widely employed to Backup data and minimize network and storage overhead by detecting and eliminating Redundancy among data. Instead of keeping multiple data copies with the same content, de duplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. De duplication has received much attention from both academia and industry because it can greatly improves storage utilization and save storage space, especially for the applications with high de duplication ratio such as archival storage systems. A number of de duplication systems have been proposed based on various de duplication strategies such as client-side or server-side de duplications, file-level or block-level de duplications. A brief review is given in Section 6. Especially, with the advent of cloud storage, data de duplication techniques become more attractive and critical for the management of ever-increasing volumes of data in cloud storage services which motivates enterprises and organizations to outsource data storage to third-party cloud providers, as evidenced by many real-life case studies. According to the analysis report of IDC, the volume of data in the world is expected to reach 40 trillion gigabytes in 2020. Today's commercial cloud storage services, such as Drop box, Google Drive and Mozy, have been applying deduplication to save the network bandwidth and the storage cost with client-side deduplication. There are two types of deduplication in terms of the size: (i) *file-level deduplication*, which discovers redundancies between different files and removes these redundancies to reduce capacity demands, and (ii) *block level deduplication*, which discovers and removes redundancies between data blocks. The file can be divided into smaller fixed-size or variable-size blocks. Using fixed size blocks simplifies the computations of block boundaries,

While using variable-size blocks (e.g., based on Rabin fingerprinting) provides better de duplication efficiency. Though de duplication technique can save the storage space for the cloud storage service providers, it reduces the reliability of the system. Data reliability is actually a very critical issue in a de duplication storage system because there is only one copy for each file stored in the server shared by all the owners. If such a shared file/chunk was lost, a disproportionately large amount of data becomes inaccessible because of the unavailability of all the files that share this file/chunk. If the value of a chunk were measured in terms of the amount of file data that would be lost in case of losing a single chunk, then the amount of user data lost when a chunk in the storage system is corrupted grows with the number of the commonality of the chunk. Thus, how to guarantee high data reliability in deduplication system is a critical problem. Most of the previous deduplication systems have only been considered in a single-server setting. However, as lots of deduplication systems and cloud storage systems are intended by users and applications for higher reliability, especially in archival storage systems where data are critical and should be preserved over long time periods. This requires that the deduplication storage systems provide reliability comparable to other high-available systems. Furthermore, the challenge for data privacy also arises as more and more sensitive data are being outsourced by users to cloud. Encryption mechanisms have usually been utilized to protect the confidentiality before outsourcing data into cloud. Most commercial storage service provider are reluctant to apply encryption over the data because it makes deduplication impossible. The reason is that the traditional encryption mechanisms, including public key encryption and symmetric key encryption, require different users to encrypt their data with their own keys. As a result, identical data copies of different users will lead to different cipher texts. To solve the problems of confidentiality and deduplication, the notion of convergent encryption has been proposed and widely adopted to enforce data confidentiality while realizing deduplication. However, these systems achieved confidentiality of outsourced data at the cost of decreased error resilience. Therefore, how to protect both confidentiality and reliability while achieving deduplication in a cloud storage system is still a challenge.

II. Related Work

Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou:Data deduplication is a technique for eliminating duplicate copies of data, and has been widely used in cloud storage to reduce storage space and upload bandwidth. Promising as it is, an arising challenge is to perform secure deduplication in cloud storage. Although convergent encryption has been extensively adopted for secure deduplication, a critical issue of making convergent encryption practical is to efficiently and reliably manage a huge number of convergent keys. This paper makes the first attempt to formally address the problem of achieving efficient and reliable key management in secure deduplication. We first introduce a baseline approach in which each user holds an independent master key for encrypting the convergent keys and outsourcing them to the cloud. However, such a baseline key management scheme generates an enormous number of keys with the increasing number of users and requires users to dedicatedly protect the master keys. To this end, we propose Dekey , a new construction in which users do not need to manage any keys on their own but instead securely distribute the convergent key shares across multiple servers. Security analysis demonstrates that Dekey is secure in terms of the definitions specified in the proposed security model. As a proof of concept, we implement Dekey using the Ramp secret sharing scheme and demonstrate that Dekey incurs limited overhead in realistic environments.

Demerits:

These works have not considered and achieved the tag consistency and integrity in the construction.

Related work

Jan Stanek, Alessandro Sorniottiy, Elli Androulakiy, and Lukas Kencl:Recent years have witnessed the trend of leveraging cloud-based services for large scale content storage, processing, and distribution. Security and privacy are among top concerns for the public cloud environments. Towards these security challenges, we propose and implement, on OpenStack Swift, a new client-side deduplication scheme for securely storing and sharing outsourced data via the public cloud. The originality of our proposal is twofold. First, it ensures better confidentiality towards unauthorized users. That is, every client computes a per data key to encrypt the data that he intends to store in the cloud. As such, the data access is managed by the data owner. Second, by integrating access rights in metadata file, an authorized user can decipher an encrypted file only with his private key.

Demerits: This scheme do not consider data privacy

R. D. Pietro and A. Sorniotti,

Deduplication is a technique used to reduce the amount of storage needed by service providers. It is based on the intuition that several users may want (for different reasons) to store the same content. Hence, storing a single copy of these files is sufficient. Albeit simple in theory, the implementation of this concept introduces many security risks. In this paper we address the most severe one: an adversary (who possesses only a fraction of the original file, or even just partially colluding with a rightful owner) claiming to possess such a file. The paper's contributions are manifold: first, we introduce a novel Proof of Ownership (POW) scheme that has all features of the state-of-the-art solution while incurring only a fraction of the overhead experienced by the competitor; second, the security of the proposed mechanisms relies on information theoretical (combinatoric) rather than computational assumptions; we also propose viable optimization techniques that further improve the scheme's performance. Finally, the quality of our proposal is supported by extensive benchmarking. Categories and Subject Descriptors H.3.5 [Information Systems]: Information storage and retrieval—Online information services

Demerits: they did not address how to minimize the key management overhead.

Module 1: Secret Sharing

Secret sharing (also called secret splitting) refers to methods for distributing a secret amongst a group of participants, each of whom is allocated a share of the secret. The secret can be reconstructed only when a sufficient number, of possibly different types, of shares are combined together; individual shares are of no use on their own.

In one type of secret sharing scheme there is one dealer and n players. The dealer gives a share of the secret to the players, but only when specific conditions are fulfilled will the players be able to reconstruct the secret from their shares. The dealer accomplishes this by giving each player a share in such a way that any group of t (for threshold) or more players can together reconstruct the secret but no group of fewer than t players can. Such a system is called a (t, n)-threshold scheme (sometimes it is written as an (n, t)-threshold scheme).

Shamir's Secret Sharing algorithm:

Shamir's Secret Sharing is an algorithm in cryptography created by Adi Shamir. It is a form of secret sharing, where a secret is divided into parts, giving each participant its own unique part, where some of the parts or all of them are needed in order to reconstruct the secret.

Counting on all participants to combine the secret might be impractical, and therefore sometimes the threshold scheme is used where any k of the parts are sufficient to reconstruct the original secret.

The secret sharing protocol

Goal: To share the secret *s* among players $P_1, P_2, \ldots, P_n$ such that *t* players are required to reconstruct the secret.

1. Dealer *D* creates a random polynomial *f*(*x*) of degree *t*-1 and constant term *s*.

$$f(x) = a_0 + a_1 x + \square + a_{t-1} x^{t-1}$$

This polynomial is constructed over a finite field, such that the coefficient a_0 is the secret *s* and all other coefficients are random elements in the field; the field is known to all participants.

2. Dealer *D* publicly chooses *n* random distinct evaluation points: $X_j \neq 0$, and secretly distributes to each player P_j the share $share_j(s) = (X_j, f(X_j))$,*j=1…n*. (Remark: The evaluation point X_j could be any publicly known value, therefore for our convenience, we assume $X_j = j$, hence the shares are denoted as $f(1),...,f(j),...,f(n)$)

Reconstruction protocol

Goal: To reconstruct the secret from each subset of *t* shares out of *n* shares. Without loss of generality we will mark this subset: $f(1),...,f(t)$

1. Use Lagrange interpolation to find the unique polynomial *f*(*x*) such that $\deg f(X) < t$ and $f(j) = share_j(s)$ for *j=1,2,..t*
2. Reconstruct the secret to be *f(0)*.

Interpolation Property*:* Given t pairs of $(i, f(i))$, with *i*'s all distinct, there is a unique polynomial *f(X)* of degree *t-1*, passing through all the points. This polynomial can be effectively computed from the pairs $(i, f(i))$.

Lagrange interpolation:

$$f(x) = \sum_{i=1}^{t} f(i) * L_i(X) \qquad \text{Where } L_i(X) \text{ is the Lagrange polynomial:} \qquad L_i(X) = \frac{\prod_{j \neq i}(x - x_j)}{\prod_{j \neq i}(x_i - x_j)}$$

Which has value 1 at X_i, and 0 at every other X_j.

Note: in the following sections the terms shareholder and server can be interchanged.

Graphic-representation of a degree-2 polynomial and its shares

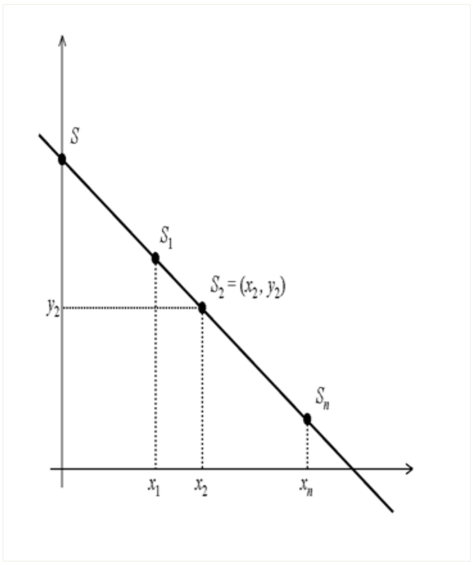

figure 1 Shamir's secret sharing scheme

Properties of Shamir's Secret Sharing:

1. **Perfect Security –** information theoretic security. Given any *t* shares, the polynomial is **uniquely determined** and hence the secret a_0 can be computed. However, given *t*-1 or fewer shares, the secret can be any element in the field and thus those shares do not supply any further information regarding the secret.

2. **Ideal –** Each share is exactly the same size as the secret.

Extendable – additional shares may easily be created, simply by calculating the polynomial in additional points.

3. **Flexible –** can assign different weights (by the number of shares) to different authorities.

Homomorphic[1] ***property* –** Shamir's secret sharing scheme has the (+,+) homomorphism property. For example, assume there are two secrets: *S, T*;are both hidden using Shamir's secret sharing scheme. Their corresponding shares are: $(f(1),....,f(n))$ which define the polynomial *f(X),* and $(g(1),...,g(n))$ which define the polynomial *g(X).* Assume further that each *i*'th shareholdersums: $h(i) = f(i) + g(i)$ ($i \in [1...n]$); then each

4.2.2 Tag Generation Algorithm.

In our constructions 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*.

4.2.3 Message Authentication Code (MAC)

One technique involves the use of a recent key to generate a small block of data, known as a message authentication code (MAC) that is appended to the message. In this technique, the two communicating parties, Alice and Bob, share a common recent key $K_{AB.}$ Alice calculates the MAC as a function of the message and the key:

$MAC_M=f(K_{AB},M)$

The message plus this MAC code are transmitted to the intended recipient. The recipient performs the same calculation on the received message, using the same recent key, to generate a new MAC code. The received MAC code is compared to the calculated code. If they match, then

a) The receiver is assured that the message has not been altered.

b) The receiver is assured that the message is from the alleged sender.

c) If the message includes a sequence number, then the receiver can be assured of the proper sequence. This is shown in Figure

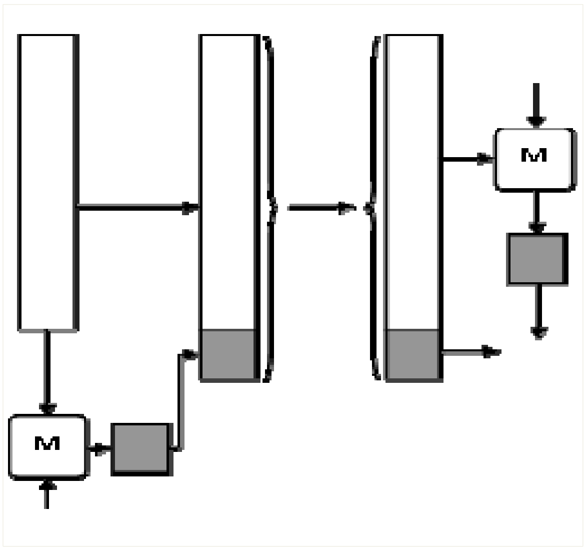

Figure 5 Message Authentication Using a Message Authentication Code(MAC)

Note 1 – A number of algorithms could be used to generate the MAC code. The NIST, in its publication entitled DES Modes of Operation, recommends the use of Data Encryption Algorithms (DEA). This algorithm is used to generate an encrypted version of the message, and only the last number of lists of ciphertext is used as the MAC code. A 16-bit or 32-bit code is typical.

Note 2 – The process just described is similar to encryption. One difference is that the authentication algorithms need not be reversible, as it must for decryption.

Note 3 – The message authentication code is also known as data authentication code (DAC).

References

1. Amazon, "Case Studies," https://aws.amazon.com/solutions/casestudies/# backup.
2. J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far 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 from duplicate 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.

AUTHOR(S) PROFILE

Dr. K. Swathi, obtained her under-graduation in B.E., (Computer Science & Engineering) from Bharathidasan University, Trichy in 1999. She obtained her M.E. degree in Computer and Communication Engineering from Anna University, Chennai in 2004.She obtained Ph.D degree in Faculty of Information & Communication Engineering from Anna University, Chennai in 2014. Presently, she is working as Associate Professor in Computer Science & Engineering, Cauvery College of Engineering and Technology, Trichy in 2008. She has 14 years teaching experience and also she had attended many workshops, seminars and conferences on Research issues in Image processing. She has published papers in international journals and presented papers in various Conferences. She is the life member of Indian Society for Technical Education (ISTE). Her areas of interest include Image processing, Data mining, network security and Software engineering

P. Vijayalakshmi, I received my B.E in Computer Science & Engineering from Arunai Engineering College, Thiruvannamalai. M.E in Computer Science & Engineering from Cauvery College of Engineering And Technology,Trichy Area of Interest:Cloud Computing and Networks.