International Journal of Scientific Research and Modern Education (IJSRME)
ISSN (Online): 2455 – 5630
(www.rdmodernresearch.com) Volume I, Issue I, 2016

SECURE DISTRIBUTED DEDUPLICATION SYSTEM IN MULTIPLE SERVERS WITH EFFICIENT AND RELIABLE KEY MANAGEMENT

Ranibala Ahongsngbam* & R. Selvakumar**
* PG Scholar, Department of Master of Computer Applications, Dhanalakshmi Srinivasan Engineering College, Perambalur, Tamilnadu
** Assistant Professor, Department of Master of Computer Applications, Dhanalakshmi Srinivasan Engineering College, Perambalur, Tamilnadu

Abstract:

Data deduplication is a technique for eradicate reproduction copy of data and has been extensively used in storage multiple server to reduce storage liberty and upload bandwidth. Even though encryption for comprehensively implement for secure deduplication a dangerous problem of making encryption realistic is to professionally and dependably handle a huge number of convergent keys. In this paper, we make a proposed system that makes the first effort to formally address the problem of accomplish efficient and dependable key management in secure reduplicate. It first establish a baseline come within reach of user data an independent master convergent key for convergent encrypting the convergent keys and outsourcing them to the Multiple Server. However, deduplication while improving storage and bandwidth efficiency is incompatible with conservative encryption. Particularly established encryption involve different users to encrypt their data with the own keys. Consequently matching data copies of different users will lead to different cipher texts making reduplication impracticable.

Key Words: Deduplication, Encryption, Storage, Server & Message Authentication Code (MAC)

1. Introduction:

In this proposed system we are going to use data deduplication process. First of all we must know what data deduplication is, it reduces the amount of data that needs to be physically stored by eliminating extra information and replacing after repetition of it with a pointer to the original. In data deduplication we remove unwanted copy of data and save the memory space. With the help of data deduplication method system reliability is improved as well as it avoids wastage of memory space of storage. With the exquisite growth of digital data, deduplication techniques are widely used to backup data and minimize network and storage overhead by detecting and removing unwanted among data. Instead of keeping multiple data copies with the same content in storage, deduplication eliminates unwanted data by keeping only one physical copy and referring other unwanted data to that copy. Deduplication has received much attention from both academic and industry because it can more improves storage utilization and save storage space, especially for the applications with high deduplication ratio such as accession storage systems. For eliminating duplicate copies of data we use data deduplication technique. To reduce storage space and for uploading bandwidth mostly it has been used, in cloud storage. A various deduplication systems has been proposed based on number of policies such as client-side or server-side deduplications, file-level or block-level deduplications. The first attempt to describe the notion of distributed safe deduplication system. The main objective of this proposed system is to describe the notion of distributed reliable deduplication system with more security. It implements new distributed deduplication system, which has more reliability. In that data chunks are distributed across multiple cloud servers. Deduplication technique can save the memory space for the cloud storage service providers; it reduces the reliability of the system. Security analysis indicates that our deduplication systems are secure in terms of the definitions specified in this security model [1]. As a proof of concept, we implement the proposed systems that indicate the acquired aerial is very limited in actual environments. Deduplication process improves storage utilization and it saves storage space. That's the reason it is useful in industry as well as in academic. It is useful in such application which has high deduplication ratio like as archival storage system. Furthermore, for the data privacy challenge is also arises more. The more sensitive data are redistributed by the users to cloud. Encoding have been usually Utilized, for to provide protection confidentiality before the redistributed data into cloud. Most commercial storage number of service providers are oppose to apply encryption over the data because it is not possible to make deduplication. The reason of that is the traditional encryption mechanism. In which including the public key encryption and symmetric key encryption have require number of users to encrypt their data with own key. For the result of similar data copy of the number of users will lead to the different data has been encrypted. To solve the problems of confidentiality and deduplication, for solving the problem of deduplication we implement notation of the convergent encryption.

2. Proposed System:

The proposed system is aim to provide more reliable and secure in deduplication process by using convergent encryption. In this proposed system, four new secure deduplication systems are proposed to provide efficient deduplication with high reliability for file-level and block-level deduplication, respectively. The secret splitting technique, instead of traditional encryption methods, is utilized to protect data confidentiality. Specifically, data are split into fragments by using secure secret sharing schemes and stored at different servers. This proposed constructions support both file-level and block-level deduplications. Security analysis demonstrates that the proposed deduplication systems are secure in terms of the definitions specified in the proposed security model.

In more details, confidentiality, reliability and integrity can be achieved in our proposed system as we are using Tag generation and a message authentication code (MAC) process in this system. Two kinds of collusion attacks are considered in our solutions. These are the collusion attack on the data and the collusion attack against servers. In particular, the data remains secure even if the adversary controls a limited number of storage servers.

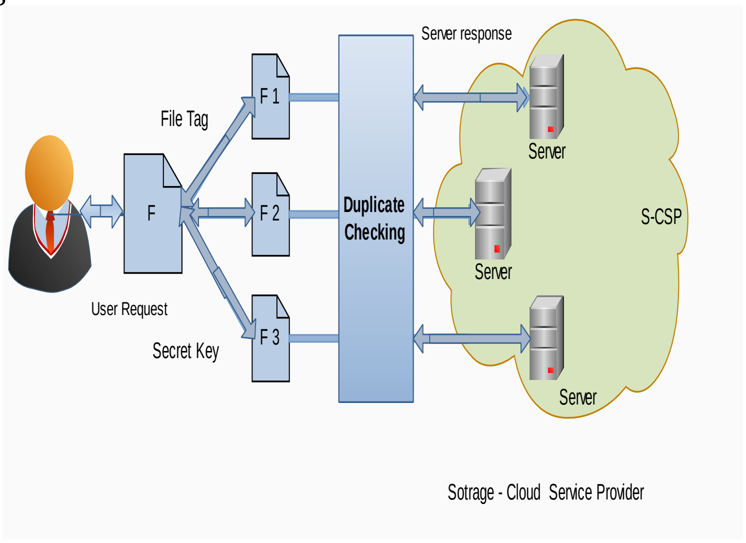

Figure 1: System Architecture

The above figure 1 shows the architecture of our proposed system, it shows the communication between the storage and the user and the encryption processes that done by server. In this system many encryption, key exchange and secrete sharing scheme is done. To give reliable and secure deduplication this convergent encryption is done for all data stored in storage, we mainly prefer cloud storage for our proposed system that provide more space and available at any time when we want to use also its advantage is large size and we can use our proposed system effectively [14]. However, the storage can be any type as our work is to do the deduplication process in any storage type to bring reliable and to save data storage with secure.

3. System Description:

In this Proposed System we used four main services to bring out our deduplication process success; these are the key features of this system and also providing some algorithm for the implementation of this system.

A. System Features:

These features of the proposed system is performed together for deduplication

- ✓ Storage-Cloud Service Provider
- ✓ Secrete sharing scheme
- ✓ Secure Deduplication System
- ✓ The File-level / The Block-level Distributed Deduplication system

Storage - Cloud Service Provider (S-CSP): In this storage cloud service provider, we used to develop cloud service and it is an entity that provides a data storage service in public cloud. The S-CSP provides the data outsourcing service and stores data on behalf of the users. To reduce the storage cost, the S-CSP eliminates the storage of redundant data via deduplication and keeps only unique data. In this paper, to assume that S-CSP is always online and has abundant storage capacity and upload bandwidth.

Secrete Sharing Scheme: In this feature, two algorithms are used which are Share and Recover. Share algorithm is used for partitioned and shared secret. With sufficient shares, extracted and retrieved the secret with the help of Recover algorithm. Share divides secret S into (k-r) fragments of same sizes, which produces r for random fragments of the equal size, and translated into simple language the k fragments using a non-systematic k-of–n erasion code into n shares of the similar size. Out of n shares the Recover adopts k from n shares as inputs. After that outputs the original secret S. A message authentication code (MAC) is a small section of knowledge used to authenticate a message and to provide integrity and authenticity certainty on the message. In our structure, the MAC is applied to derive the bonafides of the external sourced stored files.

File-Level / Block-Level Distributed Deduplication System: It support capable duplicate check, tags for each file will be calculated and send to storage cloud service provider. To prevent alignment invasion organized by the S-CSPs, tag collected at different storage servers. System Setup: In our structure, the storage cloud service provider is considered to be n with identities denoted by id1, id2,..., idn respectively. To upload file F, the client communicate with S-CSPs to perform the elimination of duplicate data. For downloading file F, the client downloads the secret shares of the file from k out of storage servers. In this part, it shows to appear how to derive the fine grained block level distributed deduplication. In this system, the client also demands to perform the file level deduplication before uploading file. The user partition this files into blocks, if no duplication is found and performs block-level deduplication. The system set up is similar to file-level deduplication and also block size parameter will be defined.

Secure Deduplication System: Consider several types of privacy need protect, that is, Unforgeability of duplicate-check token: There are two types of adversaries, that is, external adversary and internal adversary. As shown below, the external adversary can be viewed as an internal adversary without any privilege.

B. Algorithm Distribution:

The distributed deduplication systems' proposed aim is to reliably store data in the cloud while achieving confidentiality and integrity. Its main goal is to enable deduplication and distributed storage of the data across multiple storage servers. Instead of encrypting the data to keep the confidentiality of the data, our new constructions utilize the secret splitting technique to split data into shards. These shards will then be distributed across multiple storage servers.

Secret Sharing Scheme: Here 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 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 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 are 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).

- ✓ The secret can be recovered from any k or more shares
- ✓ 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). When $r = k-1$, the $(n, k, k-1)$-RSSS becomes the (n, k) Shamir's Secret Sharing Scheme (SSSS).

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. **Message Authentication Code (MAC):** A message authentication code (MAC) is a short piece of information used to authenticate a message and to provide integrity and authenticity assurances on the message. In our construction, the message authentication code is applied to achieve the integrity of the outsourced stored files. It can be easily constructed with a keyed (cryptographic) hash function, which takes input as a secret key and an arbitrary-length file that needs to be authenticated, and outputs a

MAC [7]. Only users with the same key generating the MAC can verify the correctness of the MAC value and detect whether the file has been changed or not.

MACs differ from digital signatures as MAC values are both generated and verified using the same secret key. This implies that the sender and receiver of a message must agree on the same key before initiating communications, as is the case with symmetric encryption. For the same reason, MACs do not provide the property of non-repudiation offered by signatures specifically in the case of a network-wide shared secret key: any user who can verify a MAC is also capable of generating MACs for other messages. In contrast, a digital signature is generated using the private key of a key pair, which is public-key cryptography. Since this private key is only accessible to its holder, a digital signature proves that a document was signed by none other than that holder.

Thus, digital signatures do offer non-repudiation. However, non-repudiation can be provided by systems that securely bind key usage information to the MAC key; the same key is in the possession of two people, but one has a copy of the key that can be used for MAC generation while the other has a copy of the key in a hardware security module that only permits MAC verification. This is commonly done in the finance industry

- ✓ A key generation algorithm selects a key from the key space uniformly at random.
- ✓ A signing algorithm efficiently returns a tag given the key and the message.
- ✓ A verifying algorithm efficiently verifies the authenticity of the message given the key and the tag. That is, return *accepted* when the message and tag are not tampered with or forged, and otherwise return *rejected*.

For a secure unforgivable message authentication code, it should be computationally infeasible to compute a valid tag of the given message without knowledge of the key, even if for the worst case, we assume the adversary can forge the tag of any message except the given one. Formally, A Message Authentication Code (MAC) is a triple of efficient algorithms (G, S, V) satisfying:

- ✓ G (key-generator) gives the key k on input 1^n, where n is the security parameter.
- ✓ S (Signing) outputs a tag t on the key k and the input string x.
- ✓ V (Verifying) outputs *accepted* or *rejected* on inputs: the key k, the string x and the tag t. S and V must satisfy.

$\Pr[\, k \leftarrow G(1^n),\ V(\, k, x, S(k, x)\,) = \textit{accepted}\,] = 1$

A MAC is unforgivable if for every efficient adversary A, $\Pr[\, k \leftarrow G(1^n),\ (x, t) \leftarrow A^{S(k,\cdot)}(1^n),\ x \notin \text{Query}(A^{S(k,\cdot)}, 1^n),\ V(k, x, t) = \textit{accepted}] < \text{negl}(n)$, where $A^{S(k,\cdot)}$ denotes that A has access to the oracle $S(k,\cdot)$, and $\text{Query}(A^{S(k,\cdot)}, 1^n)$ denotes the set of the queries on S made by A, which knows n. Clearly we require that any adversary cannot directly query the string x on S, since otherwise she can easily obtain a valid tag.

4. Conclusion:

The duplication of same file in storage reduce the size storage capacity and take more time to fetch our require file. Also, that may lead to make an insecure and unreliable in our process of data storing in and retrieving from storage that is uploading and downloading. In this paper, we proposed the distributed deduplication systems to improve the reliability of data while achieving the confidentiality of the users' outsourced data without an encryption mechanism. In this proposed system, four constructions were proposed to support file level and fine grained block-level data deduplication. The security of tag consistency and integrity were achieved. In this proposed system, we had implemented a deduplication system using the Ramp secret sharing scheme and demonstrated that it incurs small encoding and decoding overhead compared to the network transmission overhead in regular upload and download operations.

5. References:

1. Jin Li, Xiaofeng Chen, Xinyi Huang, Shaohua Tang and Yang Xiang Senior Member, IEEE and Mohammad Mehedi Hassan Member, IEEE and Abdulhameed Alelaiwi Member, IEEE, 2015," Secure Distributed Deduplication Systems with Improved Reliability", DOI 10.1109/TC.2015.2401017, IEEE Transactions on Computers.
2. Amazon, "Case Studies", https://aws.amazon.com/ solutions / case - studies/# backup.
3. 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 /idc- the-digital-universe-in-2020.pdf, Dec 2012.
4. M. O. Rabin, "Fingerprinting by random polynomials," Center for Research in Computing Technology, Harvard University, Tech. Rep. Tech. Report TR-CSE-03-01, 1981.
5. J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, "Reclaiming space from duplicate files in a server less distributed file system." in ICDCS, 2002, pp. 617–624.
6. M. Bellare, S. Keelveedhi, and T. Ristenpart, "Dupless: Server- aided encryption for deduplicated storage," in USENIX Security Symposium, 2013.
7. "Message-locked encryption and secure deduplication," in EUROCRYPT, 2013, pp. 296–312.
8. 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.
9. A.D. Santis and B. Masucci, "Multiple ramp schemes," IEEE Transactions on Information Theory, vol. 45, no. 5, pp. 1720–1728, Jul. 1999.
10. 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.
11. A.Shamir, "How to share a secret," Commun. ACM, vol. 22, no. 11, pp. 612–613, 1979.
12. J.Li, X. Chen, M. Li, J. Li, P. Lee, and W. Lou, "Secure deduplica- tion with efficient and reliable convergent key management," in IEEE Transactions on Parallel and Distributed Systems, 2014, pp. vol. 25(6), pp. 1615–1625.
13. 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.
14. J. S. Plank, S. Simmerman, and C. D. Schuman, "Jerasure: A library in C/C++ facilitating erasure coding for storage applica- tions - Version 1.2," University of Tennessee, Tech. Rep. CS-08-627, August 2008.
15. 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.
16. 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.
17. 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.
18. P. Anderson and L. Zhang, "Fast and secure laptop backups with encrypted de-duplication," in Proc. of USENIX LISA, 2010.
19. Z. Wilcox-O'Hearn and B. Warner, "Tahoe: the least-authority filesystem," in Proc. of ACM StorageSS, 2008.
20. 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.