Middle-East Journal of Scientific Research 24 (Techniques and Algorithms in Emerging Technologies): 558-562, 2016
ISSN 1990-9233;

DOI: 10.5829/idosi.mejsr.2016.24.TAET23639

Secure Deduplication Using De Key with Efficient and Reliable Convergent Key Management in Cloud Storage

[1]R. Thilagavathi, [2]S. Ramasamy and [3]R.K. Gnanamurthy

[1]PG Scholar, Department of Computer Science &Engineering, VCEW, Namakkal, TN, India
[2]Assistant Professor, Department of Computer Science & Engineering, VCEW, Namakkal, TN, India
[*]Professor, Department of Computer Science &Engineering, SKP, Tiruvannamalai, TN, India

Abstract: Data deduplication is a technique for reducing duplicate copies of information in cloud storage. By using Deduplication, process moment and storage space capability in cloud are decreased in efficient approach. Since it is, an occurring challenge is to execute protected de duplication in cloud storage. Plaintexts are encrypted to create convergent key and cipher text. The human or computer without using the convergent key can't read by the cipher text. Convergent key encryption is used to encrypt analogous information copies with similar cipher text and identical convergent key. We first establish a baseline approach in which every consumer grips a self-governing master key used for encrypting the convergent keys and outsourcing them to the cloud. Though, such a baseline key organization method creates a huge numeral of keys along with the growing total number of consumers and requires consumers toward contributed secure the master keys. Dekey is a new construction in which consumer do not necessitate to handle all keys on their individual except as an alternative securely allocate the convergent key distributes transversely various servers. A new-fangled enhanced Ramp secret sharing scheme and proof of Ownership (PoW) is used for dekey encryption and decryption.

Key words: Cloud Computing · Storage Space · Deduplication · Convergent Encryption · Dekey

INTRODUCTION

The motivation of the cloud storage is enterprises and organizations to outsource data storage space to intermediary cloud providers, as indicated by various real-lives glasses case studies [1]. One important challenge of now a day's cloud storage services is the organization of the growing volume of data. To formulate data administration scalable in cloud computing, deduplication has been a familiar technique and has involved more and more awareness in recent times. Data deduplication is a specific data compression technique for removing duplicate copies of reoccurring data in cloud storage space. This technique is used to improve storage space exploitation and can also be concerned to network data transmits to decrease the number of bytes that should be sent. Instead of maintaining several data copies with the similar content, deduplication removes the unnecessary data by keeping merely one material copy and referring other unnecessary data to that copy. Deduplication can obtain at the file level or the block level position. In file level deduplication, duplicate copies of the same file are removed. Deduplication can also obtain position at the block level, which removes duplicate blocks of data that happen in non-identical files. Now a day's commercial cloud storage space services, such as Mozy, Dropbox and Memopal, have been applying deduplication to consumer information to save maintenance cost.

Though data deduplication carries a lot of advantages, security [2] and privacy concerns arise as users' perceptive data are vulnerable to both inside and outside attack. Traditional encryption, as providing data confidentiality, is incompatible with data deduplication [3]. Specially, traditional encryption involves different consumer to encrypt their data with their individual keys. Therefore, the similar data copies of different users will lead to different cipher texts, creating deduplication unfeasible.

Corresponding Author: R. Thilagavathi, PG Scholar, Department of Computer Science &Engineering, VCEW, Namakkal, TN, India.

Convergent encryption [4] has been proposed to make obligatory data confidentiality while make deduplication feasible. It encrypts/decrypts a information copy with a convergent key, where this key is obtained from the cryptographic hash value computation of the content of a data copy [4]. Behind key generation and data encryption, consumers maintain the keys and send the cipher text to the cloud storage. As the encryption operation is deterministic and is obtained from the data content, identical data copies will produce the similar convergent key and hence the similar cipher text.

To avoid unauthorized access, a secure proof of ownership protocol is besides required to offer the proof that the consumer really owns the similar file while a duplicate is found. Behind the proof, consequent consumers with the similar file will be provided a pointer from the server without necessitate to uploading the similar file. A consumer can download the encrypted file with the indicator from the server, which can merely be decrypted by the corresponding data proprietors with their convergent keys. Thus, convergent encryption permits the cloud to execute deduplication on the cipher text and the proof of ownership avoids the unauthorized consumer to access the file.

To realize how convergent encryption can be recognized, we consider a baseline approach that implements convergent encryption based on a layered approach. That is, the inventive data copy is first encrypted with a convergent key obtained by the data copy itself and then encrypted by a master key that will be held in reserve locally and securely by each consumer.

The encrypted convergent keys are then store up, along with the corresponding encrypted data copies, in cloud storage space. The master key can be used to recuperate the encrypted keys and thus the encrypted files. In this approach each consumer merely requests to maintain the master key and the metadata regarding the outsourced data.

Though, the baseline approach endures two dangerous exploitation problems. First, it is inefficient, as it will create a huge number of keys with the growing total number of consumers. Specially, every consumer should correlate an encrypted convergent key with every block of its outsourced encrypted data copies; hence as to later on reinstate the data copies. While different consumers may distribute the similar data copies, they should have their individual set of convergent keys so that other users cannot access their files. Since a result, the number of convergent keys being initiated linearly balances with the number of blocks being stored and the number of consumers.

This key management overhead grows to be more important if we utilize fine-grained block-level deduplication [5]. Second, the baseline approach is unreliable, because it necessitates every consumer to obligate secure his own master key. The consumer data cannot be improved once the master key is lost; if it is concessional by attackers, then the consumer data will be issued.

Related Work

Traditional Encryption: To secure the confidentiality of outsourced information, a variety of cryptographic resolutions have been proposed in the literature [6]. Their schemes construct on traditional (symmetric) encryption, in which every consumer encrypts information with an independent secret key. A few studies [7] propose to make use of threshold secret distribution [8] to sustain the forcefulness of key management. Though, the above studies do not regard as deduplication. By means of traditional encryption, various consumers will basically encrypt the same data copies with their personal keys, but this will lead to various ciphertexts and therefore create deduplication impossible.

Proposed Work: A new construction called Dekey, which offers efficiency and reliability agreements for convergent key management on both consumer and cloud storage space regions. Our design is to be relevant deduplication to the convergent keys and force secret sharing techniques. Specially, we create secret distributes for the convergent keys and share them transversely several independent key servers. Merely the first consumer who uploads the data is necessary to calculate and share such secret shares, while all following consumers who own the identical data copy need not compute and store these shares over again. To recuperate data copies, a consumer should access a least amount of key servers through authentication and attain the secret shares to recreate the convergent keys. In other words, the secret distributes of a convergent key will merely be accessible by the authoritative consumers who own the corresponding data copy. This considerably decreases the storage overhead of the convergent keys and creates the key management reliable next to failures and attacks. In addition, the project also judges the revocation of consumers in the given group. If the original consumer of the group intimates the server with a consumer's (B) revocation, then the server rejects the proof of ownership submitted by that consumer (B).

In this Dekey construction creates the following roles.

- A new construction Dekey is proposed to offer efficient and reliable convergent key administration through convergent key deduplication and secret distributing. Dekey supports both file-level and block level deduplications.
- Protection analysis reveals that Dekey is protect in conditions of the definitions particular in the proposed security model. In particular, Dekey residue protect even the challenger controls a partial number of key servers.

Symmetric Encryption: Symmetric encryption uses a regular secret key k to encrypt and decrypt the data. Symmetric encryption methods consist of three primordial functions:

- $KeyGen_{SE}(1^{ë})$ - is the key generation algorithm that creates k using safety parameter $1^{ë}$.
- $Encrypt_{SE}(k,M)$- is the symmetric encryption algorithm that obtains the secret and information M as input and then outputs the ciphertext C.
- $Decrypt_{SE}(k,C)$- is the symmetric decryption algorithm that obtains the secret key and ciphertext C and then outputs the original information M.

Convergent Encryption: Convergent encryption [4, 9] offers data confidentiality in deduplication. A consumer (or data owner) obtains a convergent key from every original data copy and encrypts the data copy with the convergent key. In addition, the consumers obtain a tag for the data copy, such that the tag will be used to determine duplicates. Now, we presume that the tag exactness possessions clutch's, i.e., if two data copies are the identical, then their tags are similar. To identify duplicates,the consumer first sends the tag to the server side to verify if the same copy has been previously stored. Note that together the convergent key and the tag are separately obtained and the tag cannot be used to realize the convergent key and concession data confidentiality. Both the encrypted data copy and its corresponding tag will be there stored on the server side. Properly, a convergent encryption method can be defined with four primordial functions:

- $KeyGen_{CE}(M)$ - K is the key generation algorithm that plots a data copy M to a convergent key K.
- $Encrypt_{CE}(K,M)$ - C is the symmetric encryption algorithm that obtains both the convergent key K and the information M as inputs and then outputs a ciphertext C.
- $Decrypt_{CE}(K,C)$ -M is the decryption algorithm that obtains both the ciphertext C and the convergent key K as inputs and then outputs the inventive information M.
- $TagGen_{CE}(M)$ – T(M) is the tag generation algorithm that plots the inventive information M and outputs a tag T(M). We permit $TagGen_{CE}$ to create a tag from the related ciphertext [9] $T(M)=TagGen_{CE}(C)$.

Proof of Ownership: The concept of proof of ownership (PoW) is to resolve the trouble of using a small hash value as a proxy for the whole file in client-side deduplication [10], anywhere the challenger could use the storage service as a content sharing network. This proof method in PoW offers a resolution to secure the security in client-side deduplication. In this approach, a client can confirm to the server that it certainly has the file. Dekey maintains client-side deduplication with PoW to permit consumers to authenticate their possession of information copies to the storage server. Specially, PoW is executed as an interactive algorithm (denoted by PoW) execute by a prover (i.e., consumer) and a verifier (i.e.,storage server) [5]. The verifier obtains a short value ø(M) from a data copy M. To confirm the ownership of the data copy M, the prover wants to send ø' and run a proof algorithm with the verifier. It is passed if and only if ø'=ø(M) and the proof is accurate.

Ramp Secret Sharing Scheme: Dekey make use of the Ramp secret sharing scheme (RSSS) [11, 12] to store convergent keys. Specially, the RSSS creates n distributes from a secret such the secret can be improved from any k distributes but cannot be improved from less than k distributes and no data about the secret can be presumed from any r distributes. It is identified that when r = 0, the (n,K,0)-RSSS grow to be the (n,k) Rabin's Information Dispersal Algorithm (IDA) when r=k-1, the (n,k,k-1)-RSSS grow to be the (n,k) Shamir's Secret Sharing Scheme (SSSS). The (n,k,r)-RSSS makes on two primordial functions:

- *Share* splits a secret S into elements of equal size, creates r arbitrary elements of the equal size and encodes the k elements using a non-systematic k-of-n erasure code[1] into n distributes of the equal size;
- *Recover* obtains any k out of n distributes as inputs and then outputs the inventive secret S.

To create the generated distributes suitable for deduplication, we substitute the above arbitrary element with pseudorandom elements in the implementation of

Dekey. Dekey uses RSSS to offer a tunable key management method to balance along with confidentiality, reliability, storage overhead and performance.

Secure Deduplication: Data deduplication is an expert data compression technique for removing duplicate copies of replicating data. This technique is used to improve storage exploitation and can besides be applied to network data transmits to decrease the number of bytes that should be sent In the deduplication procedure, inimitable chunks of information, or byte patterns, are identify and stored during a process of psychoanalysis.

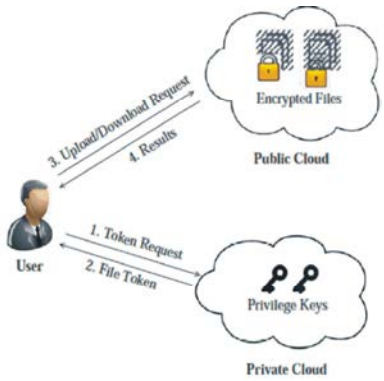
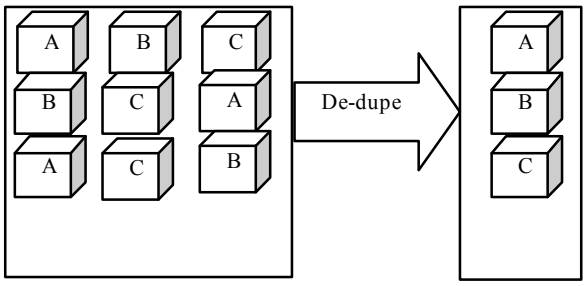


Fig. 1: Architecture for Secure Deduplication

Since the psychoanalysis maintains, other chunks are contrasted to the stored copy and at any time a match's happen, the unnecessary chunk is substitute with a small location that points to the stored chunk. Given that the identical byte pattern may happen dozens, hundreds, or even thousands of times, the amount of data that should be stored or transmitted can be really decreased.



Original data duplicates removed

Fig. 2: De-duplication

Dekey Generation: Dekey supports the both S-CSP and KM-CSP. In this Dekey generation, we leave out the normal file transmit and deduplication modules for generalization. To build full utilize of the multicore feature of modern processors, we imagine that these modules running in parallel on various cores in a pipeline approach. In the baseline approach, we basically encrypt every hash key H0 with the consumer's master key, while in Dekey, we create n distributes of H0.

We decide 4 KB as the defaulting data block size. A superior data block size (e.g., 8 KB instead of 4 KB) end results in improved encoding/decoding performance due to less chunks being handled, but has fewer storage lessening obtainable by deduplication [13, 5]. For every data block, a hash key of size 32 bytes is created using the hash function SHA-256, which belongs to the family of SHA-2. In accumulation, we agree to the symmetric-key encryption algorithm AES-256 in Cipher- Block Chaining (CBC) mode as the default encryption algorithm. Both SHA-256 and AES-256 are implemented using the EVP library of OpenSSL Version 1.0.1e.

Simulation and Results

Upload and Download: In this section, we estimated the time utilization of arbitrary records upload in the cloud, or records download from remote servers. Our method consists at first, to generate a random data file of a fixed size.

Table 1: Average time to Upload and Download file

	Average Time in "s"	
Size	Upload	Download
10	0.338	0.192
102	0.329	0.191
103	0.337	0.187
104	0.325	0.192

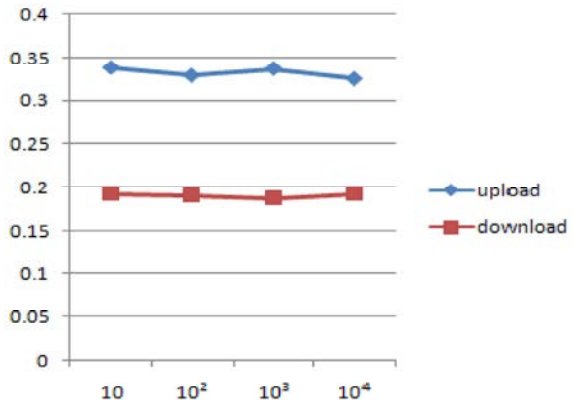


Fig. 3: Upload and Download

Then, we encrypt the records file based on the AES-CBC algorithm. The distance end to end of the used enciphering key is 256 bits (AES-256-CBC). Afterwards, we upload the encrypted data and we download it from the cloud storage space. As such, we perform a few tests

by choosing various files of various sizes. At every time, we calculated the average time for uploading and downloading the encrypted data. Next, we present the results of average time calculation to upload and download data file in the cloud storage space.

Data Confidentiality: In our representation, we propose to outsource encrypted information to remote storage space servers. That is, information is store up enciphered in the cloud, based on a symmetric encryption algorithm using for each information key. This enciphering key is content interrelated information, ensuring information deduplication [14] in remote servers. Thus, the confidentiality of outsourced information is dual. Fig. 5 shows the encoding and decoding times vs. the confidentiality level r.

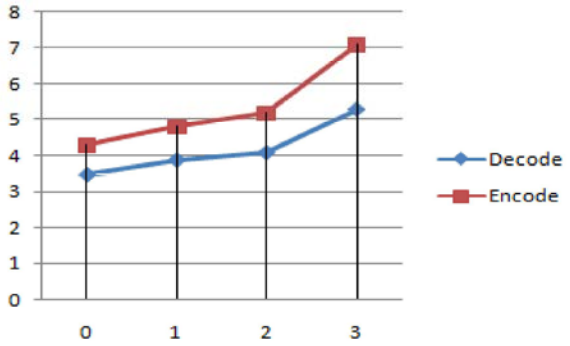


Fig. 4: Confidentiality Level

CONCLUSIONS

To overcome all the trouble now a days the utilize of cloud computing rising day by day cloud computing become a investigate topic for enhanced confidentiality and security pieces we proposed a de-duplication technique is called Dekey using Ramp Secret Sharing scheme and express that it incurs tiny encoding and decoding overhead contrasted to the network communication [15] overhead in the normal upload and download processes. Dekey applies deduplication among convergent keys and shares convergent key distributes transversely several key servers, though preserving semantic security of convergent keys and confidentiality of outsourced information.

REFERENCES

1. Amazon Case Studies.[Online]. Available: https:// aws.amazon.com/ solutions/case-studies/#backup.
2. Arulkumaran, G. and R.K. Gnanamurthy, 2014. Improving Reliability against Security Attacks by Identifying Reliance Node in MANET, Journal of Advances in Computer Networks, 2(2): 96-99.
3. Pietro, R.D. and A. Sorniotti, 2012. Boosting Efficiency and Security in Proof of Ownership for Deduplication, in Proc. ACMSymp. Inf., Comput. Commun. Security, H.Y. Youm and Y. Won, Eds., 2012, pp: 81-82.
4. Douceur, R., A. Adya, W.J. Bolosky, D. Simon and M. Theimer, 2002. Reclaiming Space from Duplicate Files in a Serverless Distributed File System, in Proc. ICDCS, 2002, pp: 617-624.
5. Ng, W.K., Y. Wen and H. Zhu, 2012. Private Data De duplication Protocols in Cloud Storage, in Proc. 27th Annu. ACM Symp.Appl. Comput., S. Ossowski and P. Lecca, Eds., 2012, pp: 441-446.
6. Gnanamurthy, RK. and L. Malathi, 2012. A novel routing protocol with lifetime maximizing clustering algorithm for WSN India Conference (INDICON), 2012 Annual IEEE925-930, December 2012.
7. Santis, A.D. and B. Masucci, 1999. Multiple Ramp Schemes, IEEE Trans. Inf. Theory, 45(5): 1720-1728.
8. Wang, W., Z. Li, R. Owens and B. Bhargava, 2009. Secure and Efficient Access to Outsourced Data, in Proc. ACM CCSW, Nov. 2009, pp: 55-66.
9. Bellare, M., S. Keelveedhi and T. Ristenpart, 2012. Message-Locked Encryption and Secure Deduplication, in Proc. IACR Cryptology ePrint Archive, 2012, pp: 296-3122012:631.
10. Halevi, S., D. Harnik, B. Pinkas and A. Shulman-Peleg, 2011. Proofs of Ownership in Remote Storage Systems, in Proc. ACM Conf. Comput. Commun. Security, Y. Chen, G. Danezis and V. Shmatikov, Eds., 2011, pp: 491-500.
11. Blakley, G.R. and C. Meadows, 1985. Security of Ramp Schemes, in Proc. Adv. CRYPTO, vol. 196, Lecture Notes in Computer Science, G.R. Blakley and D. Chaum, Eds., 1985, pp: 242-268.
12. Shamir, A., 1979. How to Share a Secret, Commun. ACM, 22(11): 612-613.
13. Clements, A.T., I. Ahmad, M. Vilayannur and J. Li, 2009. Decentralized De duplication in San Cluster File Systems, in Proc.USENIX ATC, 2009, pp: 8.
14. Meyer, D.T. and W.J. Bolosky, 2011. A Study of Practical De duplication, in Proc. 9th USENIX Conf. FAST, 2011, pp: 1-13.
15. Gnanamurthy, R.K., 2014. Design and implementation of quality of service and security in group Communication over mobile Adhoc networks.