A System for Data Reduction Using Security Factors

[1]Bedampudi .Rajesh And[2]R.Bala Dinakar
[1]PG Student, [2]Assistant Professor,
[1,2]Department of Computer Applications, Godavari Institute of Engineering and Technology, Rajahmundry, India

***Abstract*--**Cloud storage is a big factor where we can see the multiple copies of the data stored in single and mulita cloud server which increases the overheads of the cloud service providers data reduction is one of more important technique which needed to performance and storage system for increasing the efficiency of cloud computing there are many challenge which comes across for data reduction when the data storage is in secured format when they outsource the data so here we implement the secret sharing scheme with data reduction technique using convergent encryption and implement the security model for data storage with reduction system where the overheads of the system are limited in real environment in the proposed system we dived the data in fragments by using secure secret sharing schemes the proposed de-duplication systems are secure in terms of the definitions specified in the proposed security model., confidentiality, reliability and integrity

***Keywords*--** De-duplication, secret sharing, distributed storage system, reliability.

I. INTRODUCTION

By the unpredictable development of digital data, de-duplication techniques are broadly engaged to backup data and decrease network and storage transparency by notice and eradicate redundancy among data. As an alternative of maintaining multiple data copies with the same content, de-duplication reducing redundant data by maintaining only single copy and referring other redundant data to that copy. De-duplication has inward much concentration from both academic world and industry since it can really recover storage utilization and keep storage space, particularly for the applications with high de-duplication ratio such as archival storage systems. A number of de-duplication systems have been projected based on various de-duplication scheme such as clientside or server-side de-duplication, file-level or blocklevel de-duplications.Specially, with the advent of cloud storage, data de-duplication procedure grow to be more gorgeous and essential for the management of ever-increasing quantity of data in cloud storage services which inspires Endeavour and club to outsource data storage to third-party cloud providers, If we consider some of the examples as proofs:

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 is reluctant to apply encryption over the data because it makes de-duplication 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 ciphertext. To solve the problems of confidentiality and DE duplication, the notion of convergentencryption has been pro-posed and widely adopted to enforce data confidentiality while realizing DE duplication. 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 DE duplication in a cloud storage system is still a challenge.

II. LITERATURE REVIEW

A. Ramp Secret Sharing

Gives explanation of Dekey technique by using the Ramp secret sharing scheme (RSSS) to store keys. Specifically, the (n, k, r) RSSS(where n > k > r>= 0) generates n shares from a secret such that First the secret can be recovered from any k shares but cannot be recovered from fewer than k shares, and second no information about the secret can be deduced from any r shares. It is known that when r = 0, the (n, k, O) 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).

B. Secure De-Duplication:

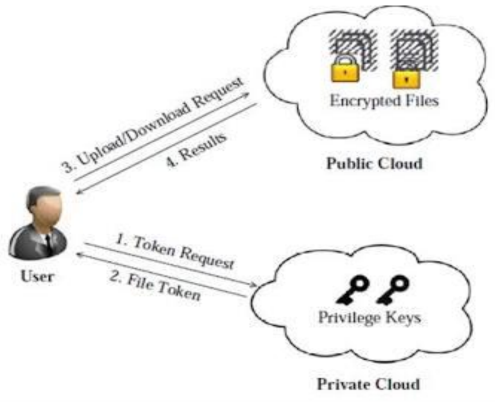
In 2008 Mark Storeet. al. developed two models for secure DE duplication storage: authenticated and anonymous. These two designs demonstrate that security can be combined with DE duplication in a way that provides a diverse range of security characteristics.

C. Proof of Ownership:

Recently, Halevi pointed out the weakness of the security in traditional de-duplication systems with only a short hashing value. Halevi showed a number of attacks that can lead to data leakage in a storage system supporting client-side de-duplication. To overcome this security issue, they also presented the concept of Proof of Ownership (PoW) to prevent these attacks. PoW enables users to prove their ownership of data copies to the storage server.

III. RELATED WORK

The various kinds of data for each user stored in the cloud and the demand of long term continuous assurance of their data safety, the problem of verifying correctness of data storage in the cloud becomes even more challenging. Cloud Computing is not just a third party data warehouse. The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc. One critical challenge of today’s cloud storage services is the management of the ever-



increasing volume of data. According to the analysis report of IDC, the volume of data in the wild is expected to reach 40 trillion gigabytes in 2020. The baseline approach suffers two critical deployment issues. First, it is inefficient, as it will generate an enormous number of keys with the increasing number of users. Specifically, each user must associate an encrypted onvergent key with each block of its outsourced encrypted data copies, so as to later restore the data copies. Although different users may share the same data copies, they must have their own set of convergent keys so that no other users can access their files. Second, the baseline approach is unreliable, as it requires each user to dedicatedly protect his own master key. If the master key is accidentally lost, then the user data cannot be recovered; if it is compromised by attackers, then the user data will be leaked.

IV. PROPOSED APPROACH

I 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. Dekey using the Ramp secret sharing scheme and demonstrate that Dekey incurs limited overhead in realistic environments we propose a new construction called Dekey, which provides efficiency and reliability guarantees for convergent key management on both user and cloud storage sides. A new construction Dekey is proposed to provide efficient and reliable convergent key management through convergent key Deduplication and secret sharing. Dekey supports both file-level Deduplication. Security analysis demonstrates that Dekey is secure in terms of the definitions specified in the proposed security model. In particular, Dekey remains secure even the adversary controls a limited number of key servers. We implement Dekey using the secret sharing scheme that enables the key management to adapt to different reliability and confidentiality levels. Our evaluation demonstrates that Dekey incurs limited overhead in normal upload/download operations in realistic cloud environments.

The advantages of placing decoys in a file system are threefold:

1. The detection of masquerade activity.
2. The confusion of the attacker and the additional costs incurred to distinguish real from bogus information, and
3. The deterrence effect which, although hard to measure, plays a significant role in preventing masquerade activity by risk-averse attackers.

V. METHODOLOGY

A. Secure De-Duplication:

Data de-duplication 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 DE duplication process, unique chunks of data, or byte patterns, are identified and stored during a process of analysis. As the analysis continues, other chunks are compared to the stored copy and whenever a match occurs, the redundant chunk is replaced with a small reference that points to the stored chunk. Given that the same byte pattern may occur dozens, hundreds, or even thousands of times (the match frequency is dependent on the chunk size), the amount of data that must be stored or transferred can be greatly reduced. This type of de-duplication is different from that performed by standard file-compression tools, such as LZ77 and LZ78. Whereas these tools identify short repeated substrings inside individual files, the intent of storage-based data de-duplication is to inspect large volumes of data and identify large sections – such as entire files or large sections of files – that are identical, in order to store only one copy of it. This copy may be additionally compressed by single-file compression techniques. For example a typical email system might contain 100 instances of the same 1 MB (megabyte) file attachment. Each time the email platform is backed up, all 100 instances of the attachment are saved, requiring 100 MB storage space.

B. User Behavior Profiling

It monitor data access in the cloud and detect abnormal data access patterns User profiling is a well known Technique that can be applied here to model how, when, and how much a user accesses their information in the Cloud. Such ‘normal user’ behavior can be continuously checked to determine whether abnormal access to a user’s information is occurring. This method of behavior-based security is commonly used in fraud detection applications. Such profiles would naturally include volumetric information, how many documents are typically read and how often. We monitor for abnormal search behaviors that exhibit deviations from the user baseline the correlation of search behavior anomaly detection with trap-based decoy files should provide stronger evidence of malfeasance, and therefore improve a detector’s accuracy.

C. Decoy Document

I propose a different approach for securing data in the cloud using offensive decoy technology. We monitor data access in the cloud and detect abnormal data access patterns. We launch a disinformation attack by returning large amounts of decoy information to the attacker. This protects against the misuse of the user’s real data. this technology to launch disinformation attacks against malicious insiders, preventing them from distinguishing the real sensitive customer data from fake worthless data the decoys, then, serve two purposes:

1. Validating whether data access is authorized when abnormal information access is detected
2. Confusing the attacker with bogus information

VI. ALGORITHM

Algorithm 1:-SystemSetup

Step 1:-The number of storage servers S-CSPs is assumed to be n with identities denoted by $id_1, id_2, ..., id_n$ respectively.
Step 2:- Define the security parameter as 1^{λ}.
Step 3:- Initialize a secret sharing scheme SS=(Share, Recover), and a tag generation algorithm TagGen..

Algorithm 2:-FileUpload

To upload a file F, the user interacts with S-CSPs to perform the de-duplication. More precisely, the user firstly computes and sends the file tag ΦF = TagGen(F) to S-CSPs for the file duplicate check.

If a duplicate is found, the user computes and sends F, id_j = TagGen′(F, id_j) to the j-th server with identity id_j via the secure channel for $1 \leq j \leq n$ and If F, id_j matches the metadata stored with F , the user will be provided a pointer for the shard stored at server id_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.

User uploads the set of values $\{\Phi F$, cj , 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.

VII. RESULT

A. Before applying the data reduction using security factor

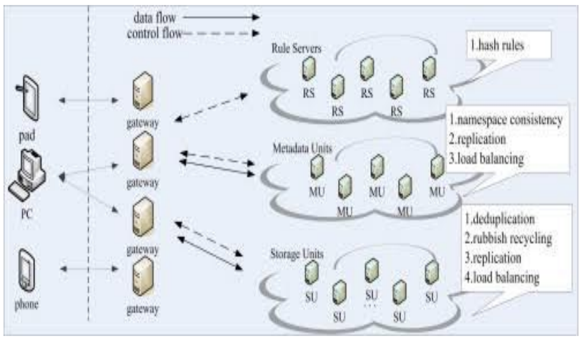


Figure 1:

B. After applying the data reduction using security factors.

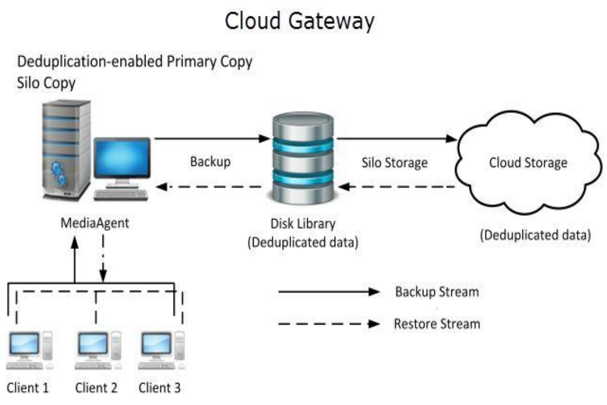


Figure 2:

CONCLUSION AND FUTURE WORK

The proposed distributed de-duplication systems are to increase the consistency of data however attaining the privacy of the user's outsourced data without an encryption appliance. The security of tag consistency and integrity ere attained. The implementation of de-duplication systems using the Ramp secret sharing scheme here gives the demonstration that it acquires small encoding/decoding overhead compared to the network transmission overhead in regular download /upload operations.

References

[1] *1+ J. Gantz and D. Reinsel, "The digital universe in 2020: Bigdigital shadows and biggest growth in the fareast," http://www.emc.com/collateral/analystreports/idcthe-digital-universe-in-2020.pdf, Dec 2012. *2+ O.

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

[3] 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.

[4] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Dupless: Server-aided encryption for deduplicated storage," in USENIX Security Symposium, 2013. *5+ "Message-locked encryption and secure DE duplication," in EUROCRYPT, 2013, pp. 296–312.

[5] G. R. Blakley and C. Meadows, "Security of ramp schemes," in Advances in Cryptology: Proceedings of CRYPTO '84, ser. Lecture Notes in Computer Science,

[6] G. R. Blakley and D. Chaum, Eds.Springer-Verlag Berlin/Heidelberg, 1985, vol. 196, pp. 242–268.

[7] *7+A. D. Santis and B. Masucci, "Multiple ramp schemes," IEEE Transactions on Information Theory, vol. 45, no. 5, pp. 1720–1728,Jul. 1999

[8] *8+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.

[9] *9+A. Shamir, "How to share a secret," Commun. ACM, vol. 22, no. 11,pp. 612–613, 1979.

[10] 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.

[11] S. Halevi, D. Harnik, B. Pinkas, and A. ShulmanPeleg, "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.

[12] 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.

[13] 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.