Data Deduplication with Secure, Distributed and Improved Reliability

Ninad Karnekar[1], Arun Gaikwad[2], Rahul Bhand[3], Yogesh Dongare[4], Prof. N.R. Pardeshi[5]
[1, 2,3 ,4, 5] Department of Computer Engineering
[1, 2, 3, 4, 5] Zeal Education society's,zeal college of engineering and research Narhe,Pune,411041

Abstract- *Data deduplication is a technique for reducing duplicate copies of data, and has been mostly used in cloud storage to reduce storage space and upload bandwidth. There is only single copy of each file stored in cloud even if such a file is owned by a huge number of users. As a result, this system improves storage utilization while reducing reliability. Further more ,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 de-duplication system. We propose new distributed de-duplication 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

As volume of data goes on increasing day by day on network, for that cloud storage system is used. But still cloud system has many challenges to face regarding storage of data. Cloud storage system provide highly available storage and parallel computing at low cost with the help of authorized access to every user. Main challenge face by cloud is storage service management of duplication. This duplication of data having wastage of storage space. to overcome this problem deduplication technique is used, which will checked duplicate copies of data; if it is found then it will eliminate these duplicate copies of data to reduce storage space and upload bandwidth. There is only one copy of data will be stored on cloud and that copy will be access by many users.

Second main challenge to cloud is security for sensitive data of user. Security requirement of data confidentiality and tag consistency. This can be achieved by introducing secret sharing in distributed storage system instead of convergent encryption. For authorized user to provide their ownership of data copies to storage system server we used POW that is proof of ownership. This is an interactive algorithm which is run by power and verifier. It is used in content distribution network, where an attacker does not know entire files but has accomplices who have file. Accomplices help attacker to obtain file, subject to constraint that they must sent fewer bits than initial min-entropy of file to attacker.

Also for privacy and security purpose we introduced decoy technique. Decoy is the bogus information such as honey pots, honey file or documents that can be generated on demand and serve as information of while detecting on unauthorized access .And also provide poison to ex-filtrated information of thief. This information will confuse an attacker into believing that bogus information whenever an unauthorized access will be detected on cloud service. This decoy information automatically return by cloud and deliver in the form of normal information. But owner of file can be identify by reading that this is bogus information. In this way true data will be remain secure.

II. EXISTING SYSTEM

The problem of verifying correctness of data storage in the cloud becomes even more challenging, As the various kinds of data for each user stored in the cloud and the demand of long term continuous assurance of their data safety. Cloud Computing is not just a third party data warehouse. By the insertion, deletion, modification, appending, reordering, etc ,the data stored in the cloud may be frequently updated by the users. Management of the ever-increasing volume of data is one of the critical challenge of today's cloud storage services . The volume of data in the wild is expected to reach 40 trillion gigabytes in 2020 according to the analysis report of IDC. The baseline approach suffers two critical deployment issues. Firstly, It will generate an enormous number of keys with the increasing number of users, as it is inefficient. Specifically, To later restore the data copies ,each user must associate an encrypted convergent key with each block of its outsourced encrypted data copies. The user must have their own set of convergent keys so that no other users can access their files, although different users may share the same data copies.

Second, the baseline approach is unreliable, as it requires each user to dedicatedly protect his own master key. The user data cannot be recovered if the master key is accidentally lost ; the user data will be leaked ; if it is compromised by attackers. The private cloud is involved as a proxy to allow data owner/users to securely perform duplicate check with differential privileges in the data deduplication systems .Such architecture is practical and has attracted much attention from researchers. While the data operation is managed in private cloud, the data owners only outsource their data storage by utilizing public cloud .

III. DISADVANTAGESS OF EXISTING SYSTEM

Traditional encryption, while providing data confidentiality, is incompatible with data deduplication. Identical data copies of different users will lead to different cipher texts, making deduplication impossible.

IV. PROPOSED SYSTEM

In this paper, we show how to design secure deduplication systems with higher reliability in cloud computing. To provide better fault tolerance we enhance our system in security, We introduce the distributed cloud storage servers into deduplication systems. Specifically, With differential privilege keys, we present an advanced scheme to support stronger security by encrypting the file. In this way, the users who didn't having corresponding privileges cannot perform the duplicate check. Furthermore, Even collude with the S-CSP such unauthorized users cannot decrypt the cipher text. Our system is secure in terms of the definitions specified in the proposed security model is demonstrates by security analysis. To protect data confidentiality, the secret sharing technique is utilized, which is also compatible with the distributed storage systems. In more details, a file is first split and encoded into fragments by using the technique of secret sharing, instead of encryption mechanisms. These shares will be distributed across multiple independent storage servers. Furthermore, A short cryptographic hash value of the content will also be computed and sent to each storage server as the fingerprint of the fragment stored at each server to support deduplication. To compute and distribute such secret shares, Only the data owner who first uploads the data is required, in other hand to compute and store these shares any more users who own the same data copy do not need . Users must access a minimum number of storage servers through authentication, to recover data copies and to reconstruct the data obtain the secret shares. In other words, Users who having their own the corresponding data copy they can only shares of data. To provide efficient deduplication with high reliability for file-level and block-level deduplication, four new secure deduplication systems are proposed, respectively. To protect data confidentiality the secret splitting technique is utilised, instead of traditional encryption methods. Specifically, By using secure secret sharing schemes, data are split into fragments and stored at different servers. 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. ADVANTAGES OF PROPOSED SYSTEM

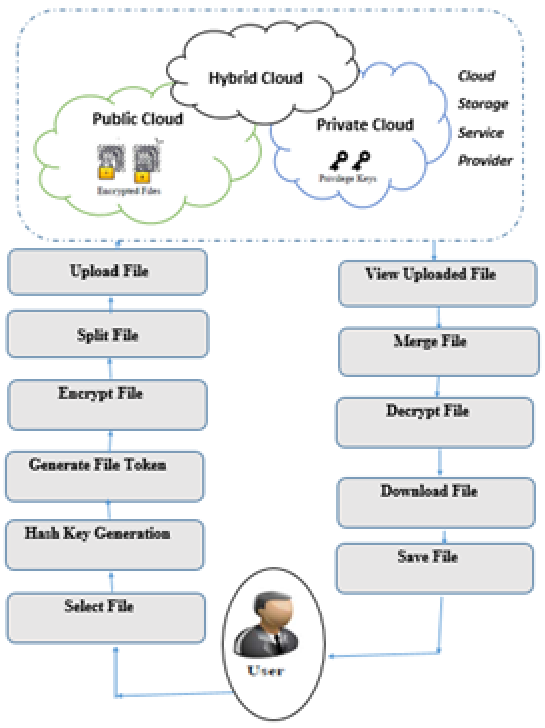
Distinguishing feature of our proposal that including tag consistency, data integrity can be achieved. To our knowledge, no existing work on secure deduplication can properly address the reliability and tag consistency problem in distributed storage systems.

Our proposed constructions support both level, file-level deduplications and block-level deduplications.

In more details, reliability, integrity and confidentiality, can be achieved in our proposed system. Security analysis demonstrates that the proposed deduplication systems are secure in terms of the definitions specified in the proposed security model. There are 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.

We implement our deduplication systems using the Ramp secret sharing scheme that enables confidentiality and high reliability levels. The redundancies are optimized and comparable with the other storage system supporting the same level of reliability and Our evaluation results demonstrate that the new proposed constructions are efficient. The user is only allowed to perform the duplicate check for files marked with the corresponding privileges. We present an advanced scheme to support stronger security by encrypting the file with differential privilege keys. Reduce the storage size of the tags for integrity check. To enhance the security of deduplication and protect the data confidentiality.

ARCHITECTURE DIAGRAM



VI. MODULES DESCRIPTION

1. Secure Deduplication file checking:

Data deduplication is a specialized data compression technique for eliminating duplicate copies of repeating data. Related and somewhat synonymous terms are single-instance (data) storage and intelligent (data) compression. To improve storage utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent by using this technique. In the deduplication process, byte patterns or unique chunks of data, are identified and stored during a process of analysis. As the analysis continues, the redundant chunk is replaced with a small reference that points to the stored chunk, other chunks are compared to the stored copy and whenever a match occurs. Given that the same byte pattern may occur hundreds, dozens, or even thousands of times (the match frequency is dependent on the chunk size), the amount of data that must be transferred or stored can be greatly reduced. Standard file-compression tools is used for this type of deduplication is different from that performed, such as LZ77 and LZ78. The intent of storage-based data deduplication is to inspect large volumes of data and identify large sections – such as entire files or large sections of files – that are identical, Whereas these tools identify short repeated sub strings inside individual files, in order to store only one copy of it. Single-file compression techniques is used for additionally compressed to this copy. 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..

2. Secret Sharing Security:

Secret sharing scheme having two algorithms, there are Recover and share. The algorithm of Recover is used for secret extracted and recovered. The secret is divided and shared by using Share in our implementation. With enough shares, 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, IEEE Transactions on Computers Volume: PP Year: 2015 any k or more shares, and (ii) no information about the secret can be deduced from any r or less shares. We will use the Ramp secret sharing scheme (RSSS) [7], [8] to secretly split a secret into shards. Two algorithms, Recover and share, are defined in the (n,k,r)-RSSS.

3. Use Behavior Profiling:

We detect abnormal data access patterns and monitor data access in the cloud 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. This method is commonly used in fraud detection applications. Such profiles would how many documents are typically read and how often, naturally include volumetric information. We monitor for the correlation of search behavior anomaly detection with trap-based decoy files should provide stronger evidence of malfeasance the abnormal search behaviors that exhibit deviations from the user baseline and therefore improve a detector's accuracy.

4. Distributed Deduplication System With Tag Consistency:

In this section, we consider maliciously-generated cipher text replacement attack or how to prevent a duplicate faking. Tag consistency is in a deduplication storage system, it requires that no adversary is able to obtain the same tag from a pair of different messages with a non-negligible probability. A security notion of tag consistency has been formalized for this kind of attack [6]. This provides security guarantees against the duplicate faking attacks in which a message can be undetectable replaced by a fake one. In the previous related work on reliable deduplication over encrypted data, the tag consistency cannot be achieved as the tag is computed by the data owner from underlying data files, which cannot be verified by the storage server.

5. Deduplication System with proof of Ownership:-

Halevi pointed out the weakness of the security in traditional deduplication 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 deduplication. PoW [12] enables users to prove their ownership of data copies to the storage server. To overcome this security issue, they also presented the concept of Proof of Ownership (PoW) to prevent these attacks. To prevent these attacks we proposed the notion of “proofs of ownership” (PoW) for deduplication systems, so that a client can efficiently prove to the cloud storage server that he/she owns a file without uploading the file itself.

6. Confidentiality:

There are two types of adversaries against confidentiality. The first type of adversary is defined as dishonest users who aim to retrieve files stored at S- CSPs they do not own. The second type of adversary is defined as a group of S-CSPs and users. To get the useful information of file content they do not own individually by launching the collusion attack is the goal of this type. The attacks launched by these two types of adversaries. The first type of adversary is denoted by Type-I attack and The second type of adversary is Type-II attack, respectively. Because the RSSS is used in our construction, the differ- ent level of confidentiality is achieved in terms of the parameter r given in the RSSS scheme, which increases with the number of r. Thus, in the following security analysis, we will not explain this furthermore.

VII. ALGORITHM

AES Algorithm steps:

The AES algorithm is used to encrypt and decrypt files which we are going to upload.

1. Derive the set of round keys from the cipher key.
   Initialize the state array with the block data (plaintext).
2. Add the initial round key to the starting state array.
3. Perform nine rounds of state manipulation.
4. Perform the tenth and final round of state manipulation.
5. Copy the final state array out as the encrypted data (ciphertext).

SHA 1 Algorithm Steps:

SHA 1 algorithm is used to hash key generation and to check file deduplication.

1. Derive the set of round keys from the cipher key.
2. Appending Padding Bits. The original message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512.
3. Appending Length. 64 bits are appended to the end of the padded message to indicate the length of the original message in bytes.
4. Preparing Processing Functions.
5. Preparing Processing Constants.
6. Initializing Buffers.
7. Processing Message in 512-bit Blocks.

HMAC SHA Algorithm steps:

The HMAC SHA algorithm is used to token generation which is used for proof of ownership

1. Append zeros to the left end of K to create a b-bit string K+(for example, if K is of length 160 bits and b = 512, then K will be appended with 44 zero bytes 0x00).
2. XOR (bitwise exclusive OR) K+withipad to produce the b-bit block Si.
3. Append M to Si.
4. Apply H to the stream generated in Step 3.
5. XOR K+withopad to produce the b-bit block So.
6. Append the hash result from Step 4 to So.
7. Apply H to the stream generated in Step 6 and output the result.

Secret Sharing Schemes (SSS):-

The secrete sharing algorithm is used for splitting and merging of uploaded file.

• The Sharing Algorithm:

Share(M) → (S1, S2, . . . , Sn, pub). The secrets S1, . . . ,Sn are dis-tributed securely among servers 1 through n, and pub is a public share. (We include pub forthe sake of generality but observe that it is often empty. If pub is present, we assume that itis authenticated, so no one can change it: it’s just published in the sky.)

• The Recovery Algorithm:

Rec(S′1, S′2, . . . , S′n, pub) = M′. The correctness property of the algorithm says that for any message M, Rec(Share(M)) = M.

VIII. RESULTS

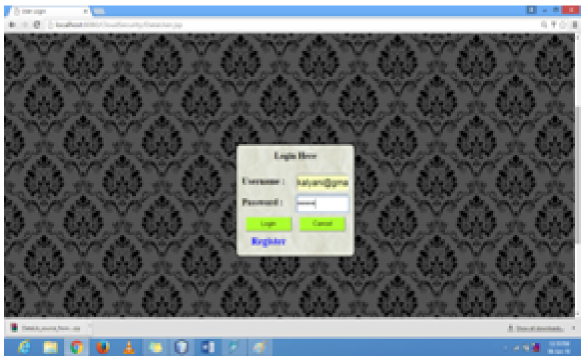


Fig: User login

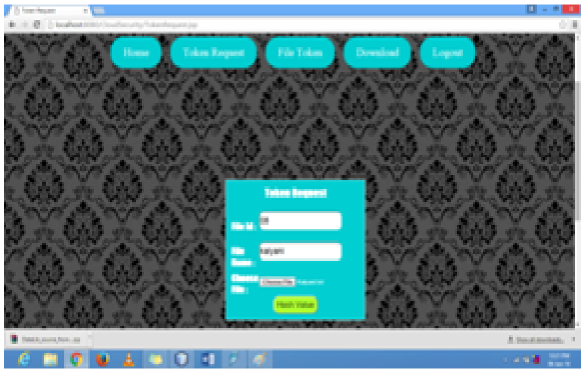


Fig: Token req

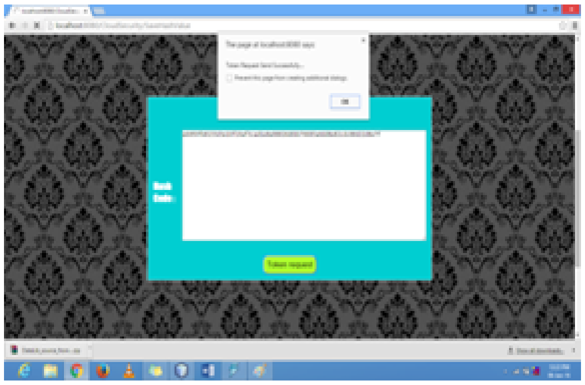


Fig: Token generation

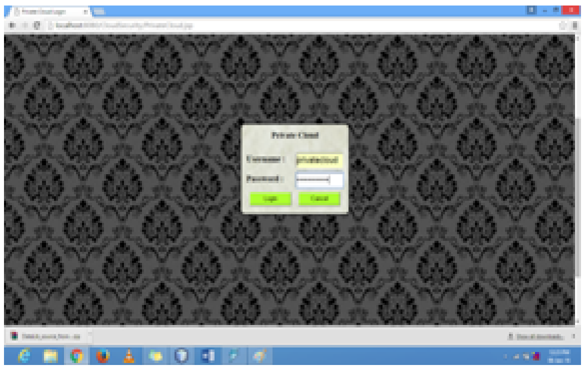


Fig: Private cloud login

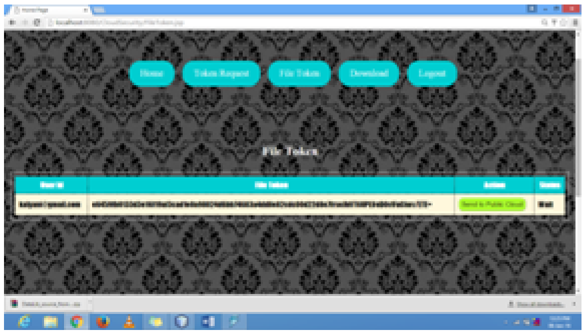


Fig: File token response from private cloud to user

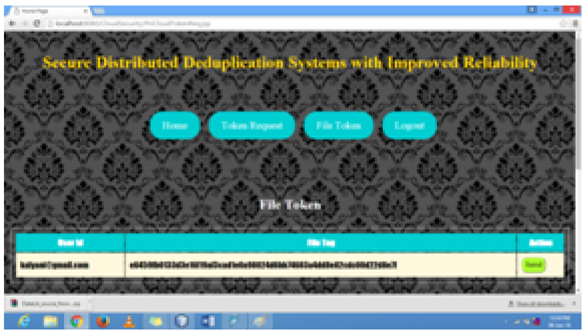

Secure Distributed Deduplication Systems with Improved Reliability
Home
Token Request
File Token
Logout
File Token
Send

Fig:File token Generation req

IX. CONCLUSION

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. Four constructions were proposed to support file-level and fine-grained block-level data deduplication. The security of tag consistency and integrity were achieved.We implemented our deduplication systems using the Ramp secret sharing scheme and demonstrated that it incurs small encoding/decoding overhead compared to the network transmission overhead in regular upload/download operations.

REFERENCE

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

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

[3] M. O. Rabin, “Fingerprinting by random polynomials,” Centerfor 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 serverlessdistributedfile system.” in ICDCS, 2002, pp. 617–624.

[5] M. Bellare, S. Keelveedhi, and T. Ristenpart, “Dupless: Serveraided encryption for deduplicated storage,” in USENIX SecuritySymposium, 2013.

[6] ——, “Message-locked encryption and secure deduplication,” inEUROCRYPT, 2013, pp. 296–312.

[7] G. R. Blakley and C. Meadows, “Security of ramp schemes,” inAdvances in Cryptology: Proceedings of CRYPTO ’84, ser. LectureNotes 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,” IEEETransactions on Information Theory, vol. 45, no. 5, pp. 1720–1728,Jul. 1999.

[9] M. O. Rabin, “Efficient dispersal of information for security, loadbalancing, 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.