ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

A Novel Approach of Encrypted Data on Cloud Storage

R.Badrinarayanan, B.V.Vahini, V.Chandrakanth, Pillai Ashwini Vijayan, Sharmila Agnal

UG Student, Dept. of CSE, SRM University, Ramapuram, Chennai, India

Assistant Professor (OG), Dept. of CSE, SRM University, Ramapuram, Chennai, India

ABSTRACT: The unstable increase of data brings new challenges to the data storage and management in cloud settings. These data classically have to be processed in a suitable style in the cloud. Thus, any superior latency may source a huge loss to the enterprises. Duplication finding plays a very major role in data management. The planned fingerprint is then comparing touching other available chunks in a database that dedicates for storing the chunks. Though, there is simply one copy for every file stored in cloud immobile if such a file is owned by a massive number of users. As a conclusion, Deduplication system improves storage consumption whereas dropping reliability. Moreover, the features of privacy for reactive data also arise while they are outsourced by users to cloud. Aiming to contract with the above safety challenges, this paper makes the first stab to honor the idea of distributed dependable Deduplication system. We propose new distributed Deduplication systems with advantaged reliability in which the data chunks are distributed diagonally various cloud servers. The security wants of data privacy and tag consistency are also achieve by introduce a deterministic furtive sharing system in distributed storage systems, as an option of using content hash encryption as in foregoing Deduplication systems.

KEYWORDS: Cloud computing; integrity auditing; Cloud storage; deduplication; file confidentiality; Cloud security.

I. INTRODUCTION

The rapid adoption of Cloud services is accompanied by increasing volumes of data stored at remote servers, so techniques for saving disk space and network bandwidth are needed. A central up and coming concept in this context is deduplication, where the server stores only a single copy of each file, regardless of how many clients asked to store that file. All clients that store the file merely use links to the single copy of the file stored at the server. According to the analysis report of IDC, the volume of data in the wild is expected to reach 40 trillion gigabytes in 2020 [1]. To make data management scalable, deduplication has been a well-known technique to reduce storage space and upload bandwidth in cloud storage. Instead of keeping multiple data copies with the same content, deduplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. Each such copy can be defined based on different granularities: it may refer to either a whole file (i.e., filelevel deduplication), or a more fine-grained fixed-size or variable-size data block (i.e., block-level deduplication). Today's commercial cloud storage services, such as Dropbox, Mozy, and Memopal, have been applying deduplication to user data to save maintenance cost [2]. However, deduplication, while improving storage and bandwidth efficiency, is incompatible with traditional encryption. Specifically, traditional encryption requires different users to encrypt their data with their own keys. Thus, same data copies of different data owners will lead to different ciphertexts, making deduplication impossible. Content hash encryption encrypts/decrypts a data copy with a convergent key, which is derived by computing the cryptographic hash value of the content of the data copy itself [3]. After key generation and data encryption, users retain the keys and send the ciphertext to the cloud. Since encryption is deterministic, identical data copies will generate the same convergent key and the same ciphertext. This allows the cloud to perform deduplication on the ciphertexts. The ciphertexts can only be decrypted by the corresponding data owners with their content hash keys.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

II.RELATED WORK

In [4] authors used the problem of integrity auditing and secure deduplication on cloud data. Specifically, aiming at achieving both data integrity and deduplication in cloud, and proposed two secure systems, namely SecCloud and SecCloud+. SecCloud introduces an auditing entity with a maintenance of a MapReduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud. Compared with previous work, the computation by user in SecCloud is greatly reduced during the file uploading and auditing phases. SecCloud+ is designed motivated by the fact that customers always want to encrypt their data before uploading, and enables integrity auditing and secure deduplication on encrypted data. 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 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.

1. The first problem is integrity auditing. The cloud server is able to relieve clients from the heavy burden of storage management and maintenance.
2. The second problem is secure deduplication. The rapid adoption of cloud services is accompanied by increasing volumes of data stored at remote cloud servers.

Considering only integrity auditing for data outsourced to cloud servers, a number of POR schemes [5], [6], and PDP schemes [7], [8], have been proposed. Among those ref.[5] has the best performance which achieves public auditing at a constant communication cost. Similar to other POR or PDP schemes, users in ref.[5] still need to perform O(k) multiplication and addition operations over the underlying field, where k is the number of checking data blocks. Batch auditing for multiple requests scenarios is not supported in ref.[5]. For secure storage deduplication, Halevi et al, introduced the first POW scheme based on the Merkle hash tree. Pietro et al. [9] enhanced ref.[10] and proposed a secure POW scheme which reduces the computational cost to a constant number of pseudorandom function operations. Nevertheless, these POW schemes do not consider data integrity auditing.

III. PROPOSED SYSTEM

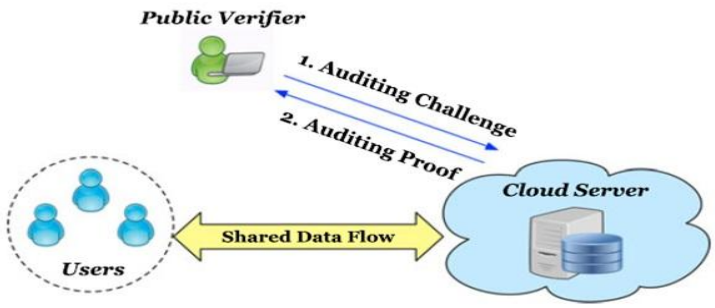

Fig.1.Architecture diagram

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

Secure Deduplication systems with higher reliability in cloud computing. Fig.1.We introduce the distributed cloud storage servers into Deduplication systems to provide better fault tolerance. To further 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.

1. New secure deduplication systems are proposed to provide efficient Deduplication with high reliability for file-level and block-level deduplication, respectively.

2. Security analysis demonstrates that the proposed deduplication systems are secure in terms of the definitions specified in the proposed security model.

3. We implement our deduplication systems using the Public secret sharing scheme that enables high reliability and confidentiality levels.

Aiming at allowing for auditable and deduplicated storage, we have three entities:

• Data Owners have large data files to be stored and rely on the cloud for data maintenance. They can be either individual consumers or commercial organizations;

• Cloud Servers virtualize the resources according to the requirements of clients and expose them as storage pools. Typically, the data owners may buy or lease storage capacity from cloud servers, and store their individual data in these spaces for future use;

• Auditor which helps clients upload and audit their outsourced data acts like a certificate authority. This assumption supposes that the auditor is associated with a pair of public and private keys. Its public key is made available to the other entities in the system.

1. *Content hash keying*

Content hash keying [12][11] provides data confidentiality in deduplication. A user or data owner derives a content hash key from the data content and encrypts the data copy with the content hash key. In addition, the content of the data copy is encoded, such that the encoded content will be used to detect duplicates. It has three primitive functions:

KeyGen(K) : The key generation algorithm takes a file content K as input and outputs the Content hash key cK of K;

Encrypt(cK,K) : The encryption algorithm takes the content hash key cK and file content K as input and outputs the ciphertext ctK;

Decrypt(cK,ctK) : The decryption algorithm takes the content hash key cK and ciphertext ctK as input and outputs the plain file K.

IV.PROPOSED ALGORITHMS

UTF-8 (UNICODE TRANSFORMATION FORMAT-8)

UTF-8 is a character encoding capable of encoding all possible characters, or code points, defined by Unicode [13]. The encoding is variable-length and uses 8-bit code units. Code points with lower numerical values, which tend to occur more frequently, are encoded using fewer bytes. The first 128 characters of Unicode, which correspond one-to-one with ASCII, are encoded using a single octet with the same binary value as ASCII, so that valid ASCII text is valid UTF-8-encoded

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

Unicode as well. Since ASCII bytes do not occur when encoding non-ASCII code points into UTF-8, UTF-8 is safe to use within most programming. Since the restriction of the Unicode code-space to 21-bit values in 2003, UTF-8 is defined to encode codepoints in one to four bytes, depending on the number of significant bits in the numerical value of the codepoint.

The salient features are as follows:

- *Backward compatibility:* One-byte codes are used for the ASCII values 0 through 127, so ASCII text is valid UTF-8. Bytes in this range are not used anywhere else, so UTF-8 text can be processed by software that can handle extended-ASCII but only applies special meaning to ASCII characters, as it will not accidentally see those ASCII characters in the middle of a multi-byte character.

- *Clear indication of byte sequence length:* The first byte indicates the number of bytes in the sequence. This makes UTF-8 a prefix code: reading from a stream can instantaneously decode each individual fully received sequence, without first having to wait for either the first byte of a next sequence or an end-of-stream indication. The length of multi-byte sequences is easily determined as it is simply the number of high-order 1s in the leading byte.

- Self-synchronisation: The leading bytes and the continuation bytes do not share values (continuation bytes start with `10` while single bytes start with `0` and longer lead bytes start with`11`). This means a search will not accidentally find the sequence for one character starting in the middle of another character. It also means the start of a character can be found from a random position by backing up at most 3 bytes to find the leading byte.

- *Sorting order:* The chosen values of the leading bytes and the fact that the continuation bytes have the high-order bits first means that sorting UTF-8 strings will produce the same order as sorting the equivalent UTF-32 strings.

AES (Advanced Encryption Standard)

AES is a block cipher with a block length of 128 bits [14]. AES allows for three different key lengths: 128, 192, or 256 bits. Most of our discussion will assume that the key length is 128 bits. Encryption consists of 10 rounds of processing for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Except for the last round in each case, all other rounds are identical. AES also has the notion of a word. A word consists of four bytes, that is 32 bits. Therefore, each column of the state array is a word, as is each row. Each round of processing works on the input state array and produces an output state array. The output state array produced by the last round is rearranged into a 128-bit output block. Like DES, AES is an iterated block cipher in which plaintext is subject to multiple rounds of processing, with each round applying the same overall transformation function to the incoming block.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

V.PERFORMANCE ANALYSIS

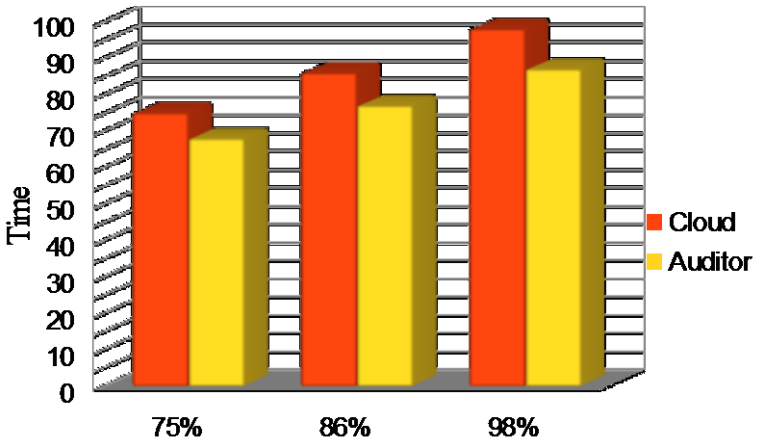

Fig.2.Integrity Auditing

Before examining the time of file auditing, we need to make analysis and identify the number of challenging blocks (i.e.,I(K)) in our integrity auditing . According to [7], if ρ fraction of the file is corrupted, through asking the proof of a constant m blocks of this file, the verifier can detect the misbehavior with probability $\alpha = 1 - (1 - \rho)m$. To capture the spirit of probabilistic auditing, we set the probability confidence α = 75%,86% and 98%, and draw the relationships between ρ and m in Fig. 2. It demonstrates that if we want to achieve low (i.e., 75%), medium (i.e., 86%) and high (i.e., 98%) confidence of detecting any small fraction of corruption, we have to respectively ask for 75,86 and 98 blocks for challenge.

VI.RESULTS

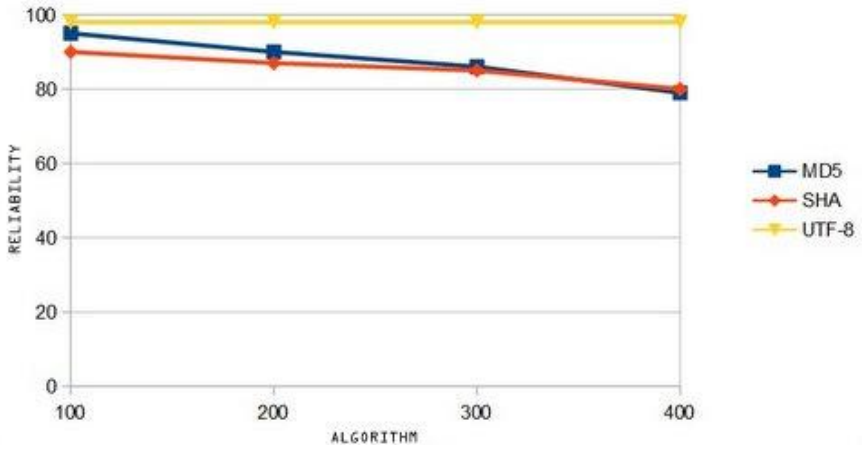

Fig.3.Reliability fraction

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering
(An ISO 3297: 2007 Certified Organization)
Website: www.ijircce.com
Vol. 5, Issue 3, March 2017

We tested the reliability of data by encoding the content using UTF-8 algorithm and using encryption and decryption with AES by applying different AES key sizes (128, 196, and 256 bits) and different data size (from 10 to 600 Mbytes). As Fig 3. shows, even when the data is large and deduplication of a data is done, the reliability remains constant. Therefore, applying UTF-8 and a symmetric encryption for data protection is a reasonable and practical choice.

VII. CONCLUSION

Compared with previous work, the computation by user is greatly reduced during the file uploading and auditing phases thereby achieving both data integrity and deduplication in cloud,. For better confidentiality and security in cloud computing we have proposed new deduplication constructions supporting authorized duplicate check in cloud architecture, in which the content of files are generated by the data owners with private keys that customers always want to encrypt their data before uploading, and allows for integrity auditing and secure deduplication directly on encrypted data. And also the proof of data owner so it will help to implement better security issues in cloud computing.

REFERENCES.

1. J. Gantz and D. Reinsel, The Digital Universe in 2020: Big Data, Bigger Digital Shadows, Biggest Growth in the Far East, Dec. 2012. [Online]. Available: http://www.emc.com/collateral/analystreports/idc-the-digital-universe-in-2020.pdf.
2. D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side Channels in Cloud Services: Deduplication in Cloud Storage," IEEE Security Privacy, vol. 8, no. 6, pp. 40-47, Nov./Dec. 2010.
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 Proc. ICDCS, 2002, pp. 617-624.
4. J. Li, J. Li, D. Xie and Z. Cai, "Secure Auditing and Deduplicating Data in Cloud," in *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2386-2396, Aug. 1 2016.doi: 10.1109/TC.2015.2389960
5. J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud," Proceedings of the ACM ASIACCS-SCC'13, 2013.
6. H. Shacham and B. Waters, "Compact proofs of retrievability," in Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security, ser. ASIACRYPT '08, Berlin, Heidelberg, May 2008, pp. 90–107.
7. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proceedings of the 14th ACM conference on Computer and communications security, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 598–609.
8. G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proceedings of the 4th international conference on Security and privacy in communication netowrks, ser. SecureComm '08. New York, NY, USA: ACM, 2008.
9. R. Di Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ser. ASIACCS '12. New York, NY, USA: ACM, 2012, pp. 81–82.
10. S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in Proceedings of the 18th ACM conference on Computer and communications security, ser. CCS '11. New York, NY, USA: ACM, 2011, pp. 491–500.
11. J. Douceur, A. Adya, W. Bolosky, P. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in 22nd International Conference on Distributed Computing Systems, 2002, pp. 617–624.
12. M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-lockedencryptionforlock-dependentmessages,"inAdvances inCryptology–CRYPTO2013,ser.LectureNotesinComputerScience, R. Canetti and J. Garay, Eds. Springer Berlin Heidelberg, 2013, vol. 8042, pp. 374–391.\
13. https://en.wikipedia.org/wiki/UTF-8#Modified_UTF-8.
14. https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture8.pdf.

BIOGRAPHY

R.Badrinarayanan is a Student in the Computer Science and Engineering Department, SRM University, Chennai, India. His research interests are Wireless Networks, Neural Networks, Genetic algorithms, etc.

B.V.Vahini is a Student in the Computer Science and Engineering Department, SRM University, Chennai, India. Her research interests are Networking, Computer Security, Cloud Computing, etc.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

V.Chandrakanth is a Student in the Computer Science and Engineering Department, SRM University, Chennai, India. His research interests are Cloud Security, Data mining, Machine learning, Operating systems.

Pillai Ashwini Vijayan is a Student in the Computer Science and Engineering Department, SRM University, Chennai, India. Her research interests are Computer Networks, Virtualization, Web Mining, etc.

Mrs. Sharmila Agnal is a Assistant Professor in the Computer Science and Engineering Department, SRM University, Chennai, India. She did her Masters in Computer Science and Engineering in Mepco Schlenk Engineering College, Sivakasi, India. Her research interests are Image processing and Data mining.