ISSN:0975-9646

An Approach for Data Integrity Checking and Data Recovery in Cloud Data Storage

Charmee V. Desai[#1] ,Prof. Gordhan B. Jethava[*2]
Computer Science & Engineering Department,
Parul Institute of Engineering & Technology
P.O. Limda, TA. Waghodia, Dist. Vadodara
Gujarat, India

Abstract-Cloud Data Storage is an attractive mean for storing and managing the outsourced data. As data are remotely stored, user is not aware of any security threat. Data modification can done by the untrusted server , unauthorized user or by some malicious activity. So user needs to be ensured that their data are intact. For this various integrity checking techniques for cloud data storage have been proposed. This paper presents an approach for checking the integrity of remotely stored data on cloud. It also uses mechanism to recover the data if integrity is violated. Proposed approach uses the metadata of constant size. It stores the metadata as the metadata of file adversary cannot access it. As file size of uploaded file is same as the original file results less communication overhead.

Keywords- Cloud Data Storage, integrity checking ,Network Coding, Data Recovery

I. INTRODUCTION

Cloud storage provides the powerful way of managing data. It allows to store and manage the data remotely. Users do not have to buy the expensive hardware and to have policies to regulate and manage the data. Apart from this users have to pay only for what they use. Because of the ubiquitous nature of the cloud, it allows to access the data from anywhere. These makes cloud storage very popular.

As cloud is distributed in nature, once user uploads the data user has no control over the data. Data are remotely stored. Users do not have physical access to the data. So user is not aware of any security threats at storage site. So to ensure the integrity of remotely stored data is major concern.Various integrity check techniques have been proposed with their pros and cons. With integrity check techniques users can be ensured that their data are intact at storage site and not have been modified by an unauthorized entity.

In 2008 Amazon has faced the downtime after that, they were not able to recover the original data.[1] In 2006 Gmail has faced the mass deletion of email which resulted the loss of data. [2] Cloud Service providers like Amazon, places explicit statements like they are responsible for any kind of data loss or data damage to save their reputation. [3] Cloud service provider are not providing any integrity check technique explicitly. They are just providing the MD5 etag value which is updated whenever file is modified remotely. So it is necessary for user to check the integrity of their remotely stored data.

This paper proposes an approach to verify the integrity of data stored at cloud storage server and to recover the data if integrity is violated.

The rest of paper is organized as follows. The second part reviews some related work. The third part describes the proposed approach. Part four is shows the experimental results and analysis. Further section presents the discussion and future concerns.

II. RELATED WORK

Many techniques for integrity checking have been proposed with their pros and cons. [4] **Provable Data Possesion (PDP) i**n this client pre-computes tags for each block of a file and stores along with file. Verification is done by generating a random challenge against a randomly selected set of file blocks without actually having to retrieve file blocks. [5]**Proof of Retrievability (PoR)** in this scheme , for the file encoded with error correcting codes sentinels are embedded for each block. Encryption is performed to make check blocks indistinguishable from other file blocks. The verifier challenges the prover by specifying the positions of a collection of sentinels. The prover returns the respective sentinels. If prover has modified or deleted a substantial portion of file, then with high probability , it will have also suppressed a number of sentinels. Using error correcting code file can be recovered. [6] **MD5 based** in this method message digest is generated from MD5 function which is used as metatdata. For verification this metadata is used. [7] **Encryption Algorithm:** In this method, At broker side a partitioner partitions the file. Tag generator module generates the tag for each segments. Hash values for each segment is calculated and stored in database. Database manager stores all relevant and required values in database. For verification, verifier retrieves all encrypted segments from the storage and calculates the hash of all segments. These values are compared with respective segment's hash value stored in database. If hash of all segments matches than file is intact otherwise tampered. [9] **RSA based:**In this, method specified is based on the RSA algorithm which makes use of large prime numbers. [8] **Generate,Encrypt and Append Metadata:** This method is based on XOR operation. First the file is divided into blocks. For each block metadata is generated by selected random bits. For Verification metadata is recalculated and compared with original data

sent by the server. Any mismatch shows the loss of integrity.[10] is similar to [8]. In this it operates on bytes rather than bits. It supports the dynamic operations with less communication overhead for both client and the server.

III. PROPOSED APPROACH

The proposed approach is similar to [] in this method integrity of only selected bytes are checked. To check the integrity of entire file keyed hash function is used which takes the file content and secrete as input and produces the message digest. This message digest is used as metadata. For data recovery Random Linear Network Coding is used.
The modified method works in following phases : Encoding, Metadata Generation, Integrity Verification and Decoding for Data Recovery. Before uploading the file ,sender encodes the file. Metadata is generated for encoded file. And then user uploads it to storage server. Verification is performed using metadata. After verification if data damage is found then decoding algorithm is executed to recover the original data.
The proposed approach works in following phases:

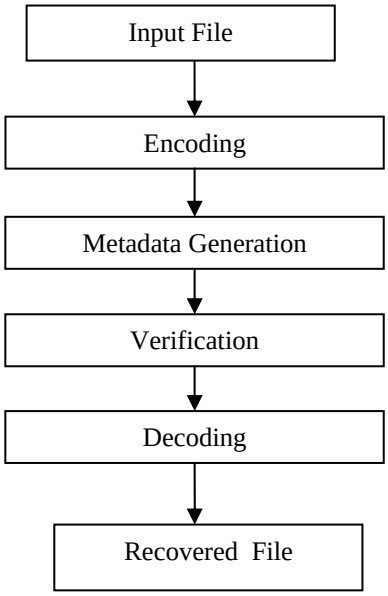

Figure 1 Phases of Proposed Work

Encoding:
For encoding purpose Random Linear Network Coding is used. Data are fetched from the file and then encoded using this technique.
Metadata Generation:
For metadata generation, first secret key is generated. This is key and message is input to HMAC function which generates the message digest which is used as metadata. Then this metadata is protected using xor function with the random alphanumeric string generated.
Verification:
In this phase, protected metadata and random alphanumeric string will be retrieved. Using the secrete key, message digest of file will be recalculated. This will be xored with the random alphanumeric string retrieved, which result in protected metadata. If retrieved and calculated protected metadata are same then file is intact otherwise file is tampered.
Decoding:
If file is found tampered , using decoding algorithm original data are recovered.
Detailed Algorithm
Encoding:
1. Take the file as input from user.
2. Apply Random Linear Network Coding on file.
For this using random function generate the random nos. Use this nos as co-efficients of matrix. Perform multiplication of this matrix with file data.

Metadata Generation
3. Generate secrete key. Store this key to client side.
To generate the secrete key ,use in built function can be used or manually key can be supplied.
4. Use this key and content of file as input to HMAC function which generates the message digest. This message digest will be used as metadata.
5. Use Random Number Generator function to generate random alphanumeric string which will be xored with generated metadata. Store this protected metadata and random alphanumeric string as metadata of file.
6. Fragment the file.

Upload
7. Upload the file.

Verification
8. For verification, recalculate the message digest using the key stored.
Retrieve stored protected metadata and alphanumeric string.
Then perform xor of alphanumeric string and calculated metadata. If received protected metadata and calculated protected metadata are same then file is intact otherwise file is tampered.

Decoding
9. If data alteration is found in verification phase, then run decoding algorithm to recover the original data.
For this first retrieve inverted matrix.
Perform multiplication of inverted matrix with data of fragments to get original data.

IV. RESULTS

The proposed approach is implemented using Amazon Storage Services. Eclipse Luna is used as editor. The java code is compiled using jdk1.8.

TABLE I
FILE SIZE AND METADATA SIZE

	File Size		
	1 KB	5 KB	10 KB
	Proposed Method	Proposed Method	Proposed Method
Size of File Uploaded	1 KB	5 KB	10 KB
Size of Metadata	32 bytes	32 bytes	32 bytes

Table 1 shows the comparison of file size of uploaded file and the size of metadata. In proposed method size of uploaded file is same as original and size of metadata is 32 bytes for all file size.

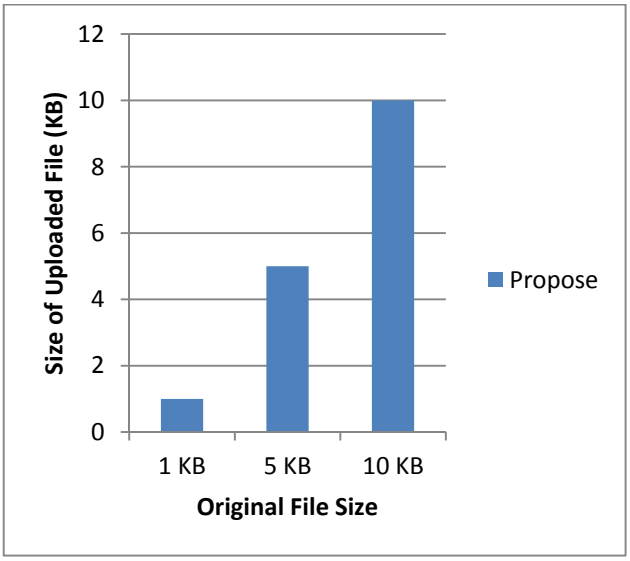

Figure 2 File Size Comparison

Fig 2 shows comparison of size of the uploaded file and original file size. It can be observed that file size is doubled when uploaded as size of metadata is same as file size and metadata is appended with the original file which increases the communication overhead.

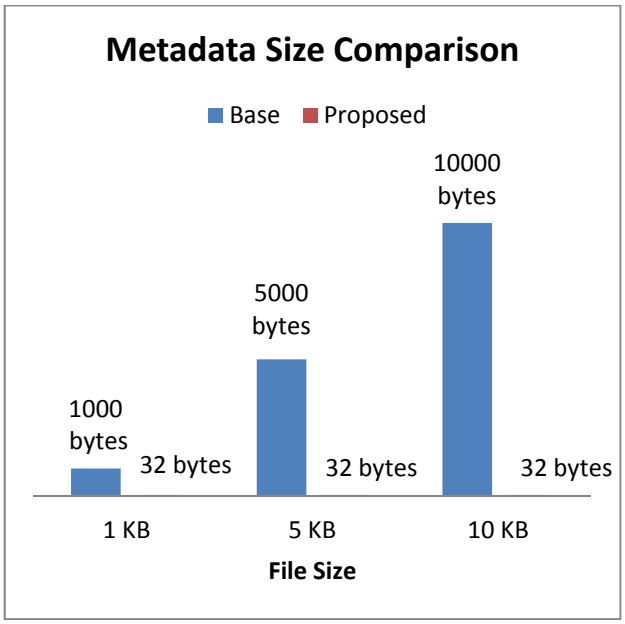

Figure 3 Metadata Size Comparison

Fig 3 shows in base method size of metadata increases with file size where as for proposed method it remains constant. As message digest produced is constant in size 256 bits (32 bytes) as HMAC SHA 256 function is used regardless of file size.

Next Time Complexity for various file Size is measured. Below chart shows the time taken to complete the entire processing.

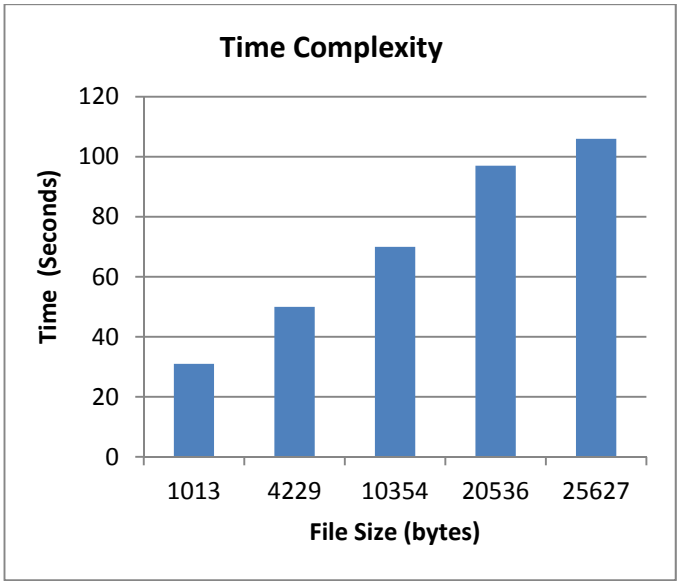

Figure 4 Time Complexity of proposed Method for various file size

Fig 4 shows the comparison of time taken for various file size. As file size increases time increases. This time includes encoding ,uploading, downloading and decoding time. Time taken also depends upon the network bandwidth and delay.

V. Conclusion And Future Work

In existing method , integrity of the remotely stored file is checked. In this method random bits are selected and used as metadata, So this method checks the integrity of the only selected bytes. If data damage found then there is no mechanism available to recover the data.

In proposed method, using the keyed hash function metadata is generated which uses the key and message itself. So using this integrity of entire file is checked. Moreover for data recovery encoding is done. Using decoding algorithm data are recovered. As file is uploaded in encoded format it provides kind of confidentiality. Size of the uploaded file is same as original file results less communication overhead compared to existing method.

Future work is **to support dynamic operations** along with integrity checking and data recovery without downloading the original file.

Acknowledgment

I would like to express my sincere thanks to my guide Prof. G.B. Jethava (Head, IT Department), for his great efforts and encouraging for this work. I am very much thankful for his continuous guidance & support. I would like to thanks all my friends and family members for their support.

References

[1] Amazon.com, “Amazon s3 Availability Event: July 20, 2008,” July 2008. http://status.aws.amazon.com/s3-20080720.html

[2] M. Arrington, “Gmail Disaster: Reports of Mass Email Deletions,” Dec. 2006. http://www.techcrunch.com/2006/12/28/gmail-disasterreports-of-mass-email-deletions/

[3] http://aws.amazon.com/agreement/

[4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable Data Possession at

Untrusted Stores," *Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07)*, pp. 598-609, 2007.

[5] A. Juels and B.S. Kaliski Jr., "Pors: Proofs of Retrievability for Large Files," *Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07)*, pp. 584-597, 2007.

[6] Sanchika Gupta,Anjani Srdan, Padam kumar,Ajit Abraham,"A Secure and Lightweight Approach for Critical Data Security in Cloud" in *Proc.CASoN-IEEE* Jan 2012

[7] Pa.Varalakshmi, Hamsavardhini Deventhiran, "Integrity Checking for Cloud Environment Using Encryption Algorithm," in *Proc. ICRTIT-IEEE* 2012

[8] Sravan Kumar R, Ashutosh Saxena, "Data Integrity Proofs in Cloud Storage," in *Proc. COMSNETS-IEEE* Jan,2011

[9] Wenjun Luo, Guojing Bai,"Ensuring The Data Integrity in Cloud Data Storage " in *Proc. CCIS-IEEE* Jan 2011

[10] Thanh Cuong Nguyen, Wenfeng Shen, Zhou Lei, Weimin Xu, Wencong Yuan, Chenwei Song" A Probabilistic Integrity Checking Approach for Dynamic Data in untrusted Cloud Storage" in *Pro. ICIS-IEEE* pp.179-183 2013