International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-2 Issue-6, August 2013

To Assure Factual Information Storage Security in Cloud Computing

Archana M, Shirisha K, Bhavani V

Abstract— Abstract-Cloud computing has evolved from virtualization, utility computing and client-server architectures and is an extension of service oriented architectures. It has been referred to as a disruptive technology which has implications on a host of issues such as licensing, scalability, cost and performance measures, privacy and security. We propose in this paper a flexible distributed storage integrity auditing mechanism, utilizing the homomorphic token and distributed erasure-coded data. Our method achieves the integrity of storage correctness guaranty and identification of misbehaving servers i.e. whenever data modifications or deletions have been detected during the storage correctness verification and error localization across cloud servers. The performance analysis shows that our scheme is more secure than existing system against Byzantine failure, unauthorized data modification attacks, and even cloud server colluding attacks.

.

Keywords-- Cloud Computing; Data Storage Security; Error Localization ; Pseudorandom Data;

I INTRODUCTION

As cloud computing continues to thrive and as more and more enterprises penetrate the cloud, security becomes a further pressing issue. Several trends are opening up the era of cloud computing, which is an internet base development and use of computer technology. The ever cheaper and more powerful processors, together with the software as a service SaaS computing architecture, are transforming data center into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centers. Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of cloud computing vendors, Amazon simple storage service (s3) and Amazon elastic compute cloud (ec2) are both well known examples. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data. Recent downtime of Amazon's s3 is such an example.

From the perspective of data security, which has always been an important aspect of quality of service, cloud computing inevitably poses new challenging security threats for number of reasons. firstly, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted due to the users' loss control of data under cloud computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of the whole data. Considering 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. Secondly, 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. to ensure storage correctness under dynamic data update is hence of paramount importance. Many companies, such as Amazon, Google, and Microsoft and so on, accelerate their paces in developing Cloud Computing systems and enhancing its services providing to a larger amount of users. Amazon's Elastic Compute Cloud (EC2) and IBM's Blue Cloud are examples of cloud computing services. These cloud service providers allow users to instantiate cloud services on demand and thus purchase precisely the capacity they require when they require based on pay- per-use or subscription-based model. Cloud users are mostly worried about the security and reliability of their data in the cloud.

Therefore, distributed protocols for storage correctness assurance will be of most importance in achieving a robust and secure cloud data storage system in the real world. However, such important area remains to be fully explored in the literature.

Fig 1 is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.

Manuscript published on 30 August 2013.
***** Correspondence Author (s)
ARCHANA M*, Computer Science and Engineering, Jawaharlal Nehru Technological University, Hyderabad, India.
SHIRISHA K , Computer Science and Engineering, Jawaharlal Nehru Technological University, Hyderabad, India.
BHAVANI V, Computer Science and Engineering, Jawaharlal Nehru Technological University, Hyderabad, India.

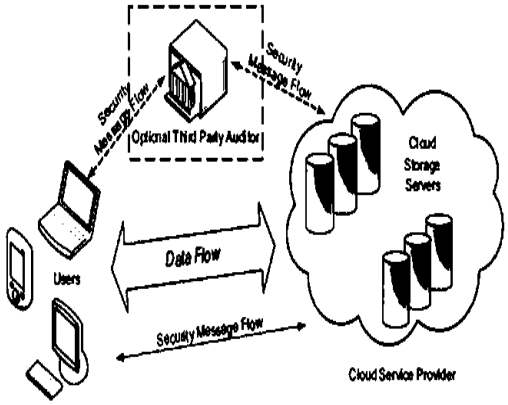

Fig.1 Cloud data storage architecture

II MODULES DESCRIPTION

1. Client Module**:**
In this module, the client sends the query to the server.

2. System Module:
Representative network architecture for cloud data storage is illustrated in Figure 1. Three different network entities can be identified as follows:
2.1 User:
Users, who have data to be stored in the cloud and rely on the cloud for data computation, consist of both individual consumers and organizations.
2.2 Cloud Service Provider (CSP):
A CSP, who has significant resources and expertise in building and managing distributed cloud storage servers, owns and operates live Cloud Computing systems,.
2.3Third Party Auditor (TPA):
An optional TPA, who has expertise and capabilities that users may not have, is trusted to assess and expose risk of cloud storage services on behalf of the users upon request.
3. Cloud data storage Module: Cloud data storage, a user stores his data through a CSP into a set of cloud servers, which are running in a simultaneous, the user interacts with the cloud servers via CSP to access or retrieve his data.

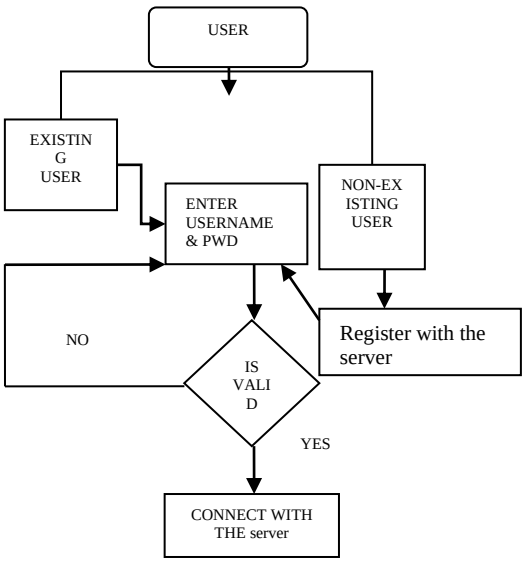

Fig. 2 Network Architecture for Cloud Data Storage

4. Cloud Authentication Server:

The Authentication Server (AS) functions as any AS would with a few additional behaviors added to the typical client-authentication protocol. The first addition is the sending of the client authentication information to the masquerading router.. The other optional function that should be supported by the AS is the updating of client lists, causing a reduction in authentication time or even the removal of the client as a valid client depending upon the request.
5. Unauthorized data modification and corruption module:
One of the key issues is to effectively detect any unauthorized data modification and corruption, possibly due to server compromise and/or random Byzantine failures. Besides, in the distributed case when such inconsistencies are successfully detected, to find which server the data error lies in is also of great significance.
6. Adversary Module:
Security threats faced by cloud data storage can come from two different sources. On the one hand, a CSP can be self-interested, untrusted and possibly malicious. On the other hand, there may also exist an economically motivated adversary, who has the capability to compromise a number of cloud data storage servers in different time intervals and subsequently is able to modify or delete users' data while remaining undetected by CSPs for a certain period. Specifically, we consider two types of adversary with different levels of capability in this paper:
6.1 Weak Adversary**:** The adversary is interested in corrupting the user's data files stored on individual servers.
6.2 Strong Adversary**:** This is the worst case scenario, in which we assume that the adversary can compromise all the storage servers so that he can intentionally modify the data files as long as they are internally consistent.

III. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

In this section, we analyze our proposed scheme in terms of security and efficiency. Our security analysis focuses on the adversary model defined in Section II. We also evaluate the efficiency of our scheme via implementation of both file distribution preparation and verification token precomputation.

A. Security Strength against Weak Adversary
1) Detection Probability against data modification*:* In our scheme, servers are required to operate on specified rows in each correctness verification for the calculation of requested Token. We will show that this "sampling" strategy on selected rows instead of all can greatly reduce the computational overhead on the server, while maintaining the detection of the data corruption with high probability.
Fig. 3, 4, 5 shows the detection probability P_d against data modification. We show P_d as a function of l (the number of blocks on each cloud storage server) and r (the number of rows queried by the user, shown as a percentage of l) for three values of z (the number of rows modified by the adversary). left) z = 1% of l; middle) z = 5% of l; right) z = 10% of l. Note that all graphs are plotted under p = 8, n_c= 10 and k = 5 and each graph has a different scale.

Suppose $n_{c:}$ servers are misbehaving due to the possible compromise or Byzantine failure. In the following analysis, we do not limit the value of $n_{c:}$, i.e., $n_{c:} \leq n$. Assume the adversary modifies the data blocks in z rows out of the l rows in the encoded file matrix.

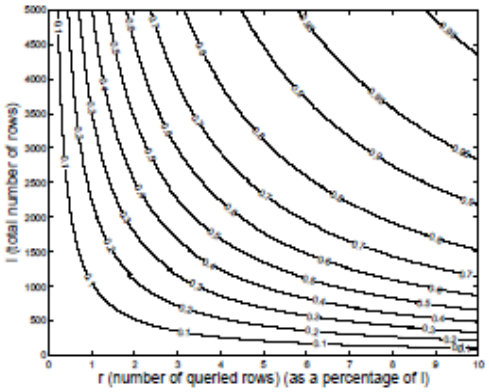

Fig. 3 Detection Probability against data modification

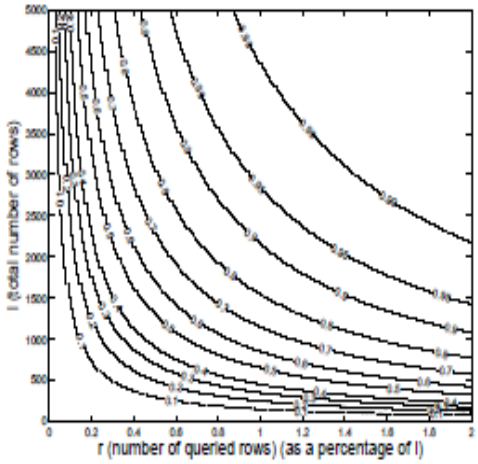

Fig. 4 Detection Probability against data modification

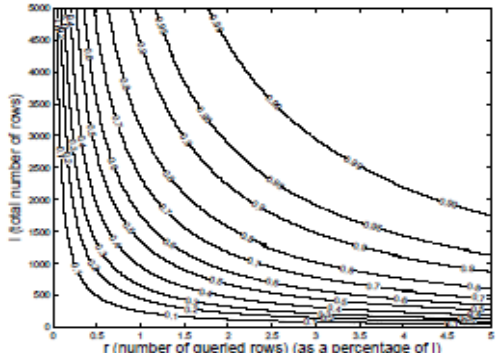

Fig. 5 Detection Probability against data modification

Let r be the number of different rows for which the user asks for check in a challenge. Let X is a discrete random variable that is defined to be the number of rows chosen by the user that matches the rows modified by the adversary. We first analyze the matching probability that at least one of the rows picked by the user matches one of the rows modified by the adversary:

$$P_m^r = 1 - p\{X=0\} = 1 - \prod_{i=0}^{r-1}\left(1 - \min\left\{\frac{z}{l-i}, 1\right\}\right)$$

$$\geq 1 - \left(\frac{l-z}{l}\right)^r .$$

Note that if none of the specified r rows in the i-th verification process are deleted or modified, the adversary avoids the detection. Next, we study the probability of a false negative result that the data blocks in those specified r rows has been modified, but the checking equation still holds. Consider the responses $R_i^{(1)}, \ldots\ldots R_i^{(n)}$ returned from the data storage servers for the i- th challenge, each response value R_i^j, calculated within GF(2^p) is based on r blocks on server j. The number of responses $R^{(m+1)}, \ldots\ldots, R^{(n)}$ from parity servers is k = n−m.Thus, according to the proposition 2 of our previous work in [14], the false negative probability is $P_f^r = P_{r1} + P_{r2}$,

Where $P_{r1} = \frac{(1+2^{-P})^{nc}-1}{2^{nc}-1}$ and $P_{r2} = (1 - P_{r1})(2^{-P})^k$

Based on above discussion, it follows that the probability of data modification detection across all storage servers is $P_d = P_m^r.\left(1 - P_f^r\right)$.

Figure 6, 7,8 plots $P_{d!}$ for different values of l, r, z while we set p = 8, $n_{c:}$ = 10 and k = 5. 2 From the figure we can see that if more than a fraction of the data file is corrupted, then it suffices to challenge for a small constant number of rows in Order to achieve detection with high probability. For example, if z = 1% of l, every token only needs to cover 460 indices in order to achieve the detection probability of at least 99%.

b. Performance Evaluation

1) File Distribution Preparation*:* We implemented the generation of parity vectors for our scheme under field GF(2^8) our experiment is conducted using C on a system with an Intel Core 2 processor running at 1.86 GHz, 2048 MB of RAM, and a 7200 RPM Western Digital 250 GB Serial ATA drive with

An 8 MB buffer. We consider two sets of different parameters for the (m + k, m) Reed-Solomon encoding. Table I shows the average encoding cost over 10 trials for an 8 GB file. In the maintain the systematic layout of data file, we table on the top, we set the number of parity vectors constant at 2. In the one at the bottom, we keep the number of the data vectors fixed at 8, and increase the number of parity vectors. Note that as m increases, the length l of data vectors on each server will decrease, which results in fewer calls to the Reed-Solomon encoder. Thus the cost in the top table decreases when more data vectors are involved.

From Table I, it can be shown that the performance of our scheme is comparable to that of [10], even if our scheme supports dynamic data operation while [10] is for static data only.

TABLE I: The cost of parity generation in seconds for an 8GB data file. For set I, the number of parity servers' k is

SetII	*K=1*	*K=2*	*K=3*	*K=4*
M=8	*358.90s*	*437.22s*	*584.55s*	*733.34s*

fixed; for set II, the number of data server's m is constant.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Set I	*M=4*	*M=6*	*M=8*	*M=10*
K=2	*567.45s*	*484.55s*	*437.22s*	*414.22s*

IV. CONCLUSION

In this paper, we investigated the problem of data security in cloud data storage, which is essentially a distributed storage system. To ensure the correctness of users' data in cloud data storage, we proposed an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. We rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. By utilizing the homomorphic token with distributed verification of erasure coded data, our scheme achieves the integration of storage correctness insurance and data error localization, i.e., whenever data corruption has been detected during the storage correctness verification across the distributed servers, we can almost guarantee the simultaneous identification of the misbehaving server(s). Through detailed security and performance analysis, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks. We believe that data storage security in Cloud Computing, an area full of challenges and of paramount importance, is still in its infancy now, and many research problems are yet to be identified. We envision several possible directions for future research on this area. The most promising one we believe is a model in which public verifiability is enforced. Public verifiability, supported in [6] [4] [17], allows TPA to audit the cloud data storage without demanding users' time, feasibility or resources. An interesting question in this model is if we can construct a scheme to achieve both public verifiability and storage correctness assurance of dynamic data. Besides, along with our research on dynamic cloud data storage, we also plan to investigate the problem of fine-grained data error Localization.

V. RELATED WORK

Juels et al. [3] described a formal "proof of retrievability"(POR) model for ensuring the remote data integrity. Their scheme combines spot-cheking and error-correcting code to ensure both possession and retrievability of files on archive service systems. Shacham et al. [4] built on this model and constructed a random linear function based homomorphic authenticator which enables unlimited number of queries and

Requires less communication overhead. Bowers et al. [5] proposed an improved framework for POR protocols that generalizes both Juels and Shacham's work. Later in their subsequent work, Bowers et al. [10] extended POR model to distributed systems. However, all these schemes are focusing on static data. The effectiveness of their schemes rests primarily on the preprocessing steps that the user conducts before outsourcing the data file F. Any change to the contents of F, even few bits, must propagate through the error-correcting code, thus introducing significant computation and communication complexity. Ateniese et al. [6] defined the "provable data possession"(PDP) model for ensuring possession of file on untrusted storages. Their scheme utilized public key based homomorphic tags for auditing the data file, thus providing public verifiability. However, their scheme requires sufficient computation overhead that can be expensive for an entire file. In their Subsequent work, Ateniese et al. [7] described a PDP scheme That uses only symmetric key cryptography. This method has lower-overhead than their previous scheme and allows for block updates, deletions and appends to the stored file, which has also been supported in our work. However, their scheme focuses on single server scenario and does not address small data corruptions, leaving both the distributed scenario and data error recovery issue unexplored. Curtmola et al. [15] aimed to ensure data possession of multiple replicas across the distributed storage system. They extended the PDP scheme to cover multiple replicas without encoding each replica separately, providing guarantee that multiple copies of data are actually maintained. In other related work, Lillibridge et al. [9] presented a P2P backup scheme in which blocks of a data file are dispersed across m+k peers using an (m+k,m)-erasure code. Peers can request random blocks from their backup peers and verify the integrity using separate keyed cryptographic hashes attached on each block. Their scheme can detect data loss from free riding peers, but does not ensure all data is unchanged. Filho et al. [16] proposed to verify data integrity using RSA-based hash to demonstrate uncheatable data possession in peer-to peer file sharing networks. However, their proposal requires exponentiation over the entire data file, which is clearly impractical for the server whenever the file is large. Shah et al. [17] proposed allowing a TPA to keep online storage honest by first encrypting the data then sending a number of precomputed Symmetric-keyed hashes over the encrypted data to the auditor. However, their scheme only works for encrypted files, and auditors must maintain long-term state. Schwarz et al. [8] proposed to ensure file integrity across multiple Distributed servers, using erasure-coding and block-level file integrity checks. However, their schemes only considers static data files and do not explicitly studies the problem of data error localization, which we are considering in this work.

REFERENCES

[1] Amazon.com, "Amazon Web Services (AWS)," Online at http://aws.amazon.com, 2008.

[2] N. Gohring, "Amazon's S3 down for several hours," Online at http://www.pcworld.com/businesscenter/article/142549/amazons s3 down for several hours.html, 2008.

[3] A. Juels and J. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files," *Proc. of CCS '07*, pp. 584–597, 2007.

[4] H. Shacham and B. Waters, "Compact Proofs of Retrievability," *Proc.of Asiacrypt '08*, Dec. 2008.

[5] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Cryptology ePrint Archive, Report 2008/175,2008, http://eprint.iacr.org/.

[6] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, "Provable Data Possession at Untrusted Stores," *Proc. Of CCS '07*, pp. 598–609, 2007.

[7] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," *Proc. of SecureComm '08*, pp. 1–10, 2008.

[8] T. S. J. Schwarz and E. L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," *Proc.* 9] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A Cooperative Internet Backup Scheme," *Proc. of the 2003 USENIX Annual Technical Conference (General Track)*, pp. 29–41, 2003.

[10] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008, http://eprint.iacr.org/.
[11] L. Carter and M. Wegman, "Universal Hash Functions," *Journal of Computer and System Sciences*, vol. 18, no. 2, pp. 143–154, 1979.]
12] J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasure coded Data," *Proc. 26th ACM Symposium on Principles of Distributed Computing*, pp. 139–146, 2007.
[13] J. S. Plank and Y. Ding, "Note: Correction to the 1997 Tutorial on Reed-Solomon Coding," University of Tennessee, Tech. Rep. CS-03-504, 2003.
[14] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," *Proc. of IEEE INFOCOM*, 2009.
[15] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-Replica Provable Data Possession," *Proc. of ICDCS '08*, pp. 411–420, 2008.
[16] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Cryptology ePrint Archive, Report 2006/150, 2006, http://eprint.iacr.org/.
[17] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," *Proc. 11th USENIX Workshop on Hot Topics in Operating Systems (HOTOS '07)*, pp. 1–6, 2007.

Archana. M currently a P.G student from TRR College of Engineering, JNTU University, Hyderabad under the supervision of Asst. Prof. V. Bhavani Initially obtained Bachelor degree from Osmania University, Hyderabad in 2008 and then Master Degree (M.C.A) from Nizam college, Osmania University, Hyderabad in 2011.Published 03 Paper in International Journals, 02 in National Conferences, attended 01 International Conference 02 National Workshops/Conferences. Having a strong interest in the design of techniques to detect the intrusions and to enhance user privacy.

Shirisha. K She is currently a P.G student from TRR College of Engineering, JNTU University, Hyderabad under the supervision of Asst. Prof. V. Bhavani . Initially obtained her Bachelor degree from Nizam college, Osmania University, Hyderabad in 2006. She attends the Nationl Conferences and the Workshops conducted at different colleges in Hyderabad.

Bhavani. V is an associate professor at the department of Computer Science and Engineering, TRR College of Engineering, Hyderabad. Initially she obtained her P.G from Indra Reddy Memorial College of Engineering and Technology, JNTU, Hyderabad. She had seven years of experience in teaching field in Computer Science Department. She is a Computer Laboratory Incharge. She Published 02 Papers in International Journals, 01 in National Conferences.She Attend Workshops in Institutions

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)