www.ijemr.net ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962

Volume-5, Issue-3, June-2015

International Journal of Engineering and Management Research

Page Number: 558-562

Security and Dependable Distributed Storage Services

B. Kiranmayee[1], S. Aarthi[2], M. Padma[3]

[1, 2, 3]Assistant Professor, CSE Department, G. Pulla Reddy Engineering College, Kurnool, A.P, INDIA

ABSTRACT

Cloud computing is the delivery of computing and storage capacity as a service to a community of end-recipients. Cloud computing entrusts services with a user's data, software and computation over a network. Cloud storage enables users to remotely store their data and enjoy the on-demand high quality cloud applications without the burden of local hardware and software management. Though the benefits are clear, such a service is also relinquishing users, physical possession of their outsourced data, which unavoidably poses new security risks toward the correctness of the data in cloud. In order to address this new problem and further achieve a secure and dependable cloud storage service, we propose in this paper a flexible distributed storage integrity auditing mechanism, utilizing the homomorphism token and distributed erasure-coded data. The propose d design allows users to audit the cloud storage with very lightweight communication and computation cost. The auditing result not only ensures strong cloud storage correctness guarantee, but also simultaneously achieves fast data error localization, i.e., the identification of misbehaving server. Considering the cloud data are dynamic in nature, the proposed design further supports secure and efficient dynamic operations on outsourced data, including block modification, deletion, and append. Analysis shows the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server scheming attacks.

Keywords*----* Cloud Computing, Dependable distributed storage, Public Auditing, Integrity

I. INTRODUCTION

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 by use of 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 Cloud data and owner's constrained computing capabilities further makes the task of data. Correctness auditing in a cloud environment expensive and even formidable for individual cloud Customers.

Therefore, enabling public audit ability or cloud storage is of critical importance so that owners can resort to a specialized third party auditor (TPA) to audit cloud storage services and maintain strong storage correctness guarantee, while saving their own precious computing resources.

We propose an effective and flexible distributed storage verification scheme with explicit dynamic data support to ensure the correctness and availability of users' data in the cloud. We rely on erasure correcting code in the file distribution preparation to provide redundancies and guarantee the data dependability against Byzantine servers where a storage server may fail in arbitrary ways. This construction drastically reduces the communication and storage overhead as compared to the traditional replication-based file distribution techniques.

By utilizing the homomorphism token with distributed verification of erasure-coded data, our scheme achieves the storage correctness insurance as well as data error localization, whenever data corruption has been detected during the storage correctness verification, our scheme can almost guarantee the simultaneous localization of data errors, i.e., the identification of the misbehaving server(s). In order to strike a good balance between error resilience and data dynamics, we further explore the algebraic property of our token computation and erasure-coded data, and demonstrate how to efficiently support dynamic operation on data blocks, while maintaining the same level of storage correctness assurance. In the real

scenario, they are renting the physical infrastructure, platforms and applications within a shared architecture. Cloud offerings can vary from virtual infrastructure, computing platforms, centralized data centers to end-user Web-Services and Web applications to enormous other focused computing services.

II. PROBLEM STATMENT

2.1 System Model

Three different network entities can be identified as follows:

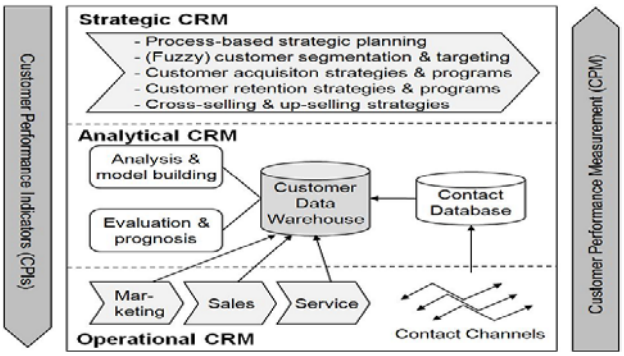
User: anentity, which has large data files to be stored in the cloud and relies on the cloud for data maintenance and computation, can be either individual consumers or organizations.

Cloud Service Provider (CSP): an entity, which is managed by Cloud Service Provider (CSP), has significant storage space and computation resource to maintain clients' data.

Third Party Auditor (TPA): loud consumers save data in cloud server so that security as well as data storage correctness is primary concern. A novel and homogeneous structure is introduced [4] to pro-security, BLS (Boneh–Lynn–Shacham) algorithm is used to signing the data blocks before outsourcing data into cloud.BLS (Boneh–Lynn–Shacham) algorithm is efficient and safer than the former algorithms. Batch auditing is achieved by using bilinear aggregate signature technique simultaneously. Reed-Solomon technique is used for error correction and to ensure data storage correctness. Multiple batch auditng is an important feature of this proposed work. It allows TPA to perform multiple auditing tasks for different users at the same.

Using Cloud Storage, users can remotely store their data and enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third party auditor (TPA) to check the integrity of outsourced data and be worry-free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy, and introduce no additional online burden to user.

In this paper, we propose a secure cloud storage system supporting privacy-preserving public auditing. We further extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient. TPA through the auditing protocol should be prohibited.



III. SYSTEM ARCHITECTURE

3.1 Controlled Data Sharing

Recently, much of growing interest has been pursued in the context of remotely stored data verification. Various sensitive data pooled in the cloud demands the cloud data sharing service to be responsible for secure, efficient and reliable enforcement of data content access among potentially large number of users on behalf of data owners. As cloud server may no longer be in the same trusted domain as the data owners, we have to rethink the problem of access control in this open environment, where cloud server takes full charge of the management of the outsourced data but are not necessarily trusted with respect to the data confidentiality. What makes the problem more challenging is the enforcement of fine-grained data access, the support of access privilege updates in dynamic scenarios, and the system scalability, while maintaining low level complexity of key management and data encryption.

At the same time, though, such a service is also eliminating data owners' ultimate control over the fate of their data, which data owners with high service-level requirements have traditionally anticipated. As owners no longer physically possess their cloud data, previous cryptographic primitives for the purpose of storage correctness protection cannot be adopted, due to their requirement of local data copy for the integrity verification. Besides, the large amount of cloud data and owner's constrained computing capabilities further makes the task of data correctness auditing in a cloud environment expensive and even formidable for individual cloud customers. Therefore, enabling public auditability [1,6] for cloud storage is of critical importance so that owners can resort to a specialized third party auditor (TPA) to audit cloud storage services and maintain strong storage correctness guarantee, while saving their own precious computing resources.

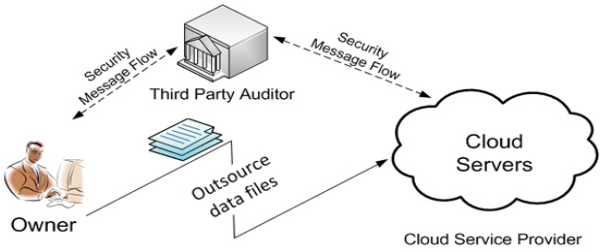


Fig.2. specialized third party auditor (TPA)

To achieve fine-grainedness, we propose to treat data as files associated with a set of meaningful attributes, use logical composition of attributes to reflect fine-grained data access, and enforce owner's control via attribute-based encryption. For the inherent scalability requirement of cloud system, where user access privilege updates happen very frequently and thus inevitably incurs significant user/data management burden on data owner, we further propose to treat the cloud as a mediated proxy, to which data owners can delegate most cumbersome workload, like handling user access privilege dynamics in large system, without affecting the underlying data confidentiality. The experiment results demonstrate the proposed scheme is highly efficient. Extensive security analysis shows our scheme is resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks. This paper presents a brief evaluation on how Cloud Computing paradigm can be used to meet the increasing demands of the Information Support Systems and how Cloud Computing paradigm can prove to be future solution for such systems.

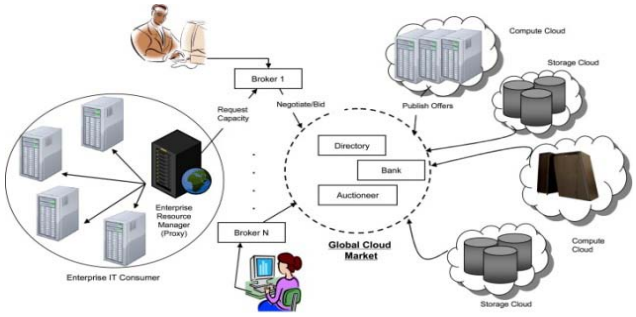


Fig.3.Attribute-based encryption

3.2 Pre-Computed Verification Tokens

To verify the correctness of user's data & to locate the errors, we entirely rely on the pre-computed verification tokens. These tokens are calculated before file distribution & they are very short. We are computing the tokens by pseudorandom function(PRF) and pseudorandom permutation function(PPF) We pre-computes short verification tokens on individual vector, Each token covering a random subset of data blocks. We have assumed block size as 256 bits & r as 8 number of verification per indices. We have three data devices and three checksum devices. Then n=3 and m=3 we choose w=4 since 2^w> n+m.

3.3 Correctness Verification

The newly computed tokens from servers for each challenge are compared with pre-computed tokens to determine the correctness of the distributed storage to eliminate the errors in storage systems key prerequisite is to locate the errors. However, many previous schemes do not explicitly consider the problem of data error localization, thus only provide binary results for the storage verification. In our scheme we integrate the correctness verification and error localization in our challenge-response protocol.. This also gives information to locate potential data errors.

3.4 File Retrieval and Error Localization

The comparison of pre-computed tokens and received response values can guarantee the identification of misbehaving server. Therefore user can recover the corrupted data. Our system recovers data from backup server & distributes all data vectors to corresponding servers. (P) ← Gen Proof (F, Φ, chal) this will results in successful recovery of corrupted data. But due to file splitting we made at the time of file distribution, user‟s need to recover file from all the servers.

IV. DISCUSSION AND ANALYSIS

We evaluate the security of the proposed scheme by analyzing its fulfillment of the security guarantee described in the storage correctness and privacy-preserving property. We start from the single user case, where our main result is originated. Then we show the security guarantee of batch auditing for the TPA in multi-user setting Algorithms are implemented using open-source erasure coding library J erasure written in C. We now assess the performance of the proposed privacy-preserving public auditing schemes to show that they are indeed lightweight. We will focus on the cost of the efficiency of the privacy-preserving protocol and our proposed batch auditing technique. The experiment is conducted using C on a Linux 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. Our code uses the Pairing-Based Cryptography (PBC) library version 0.4.18. The elliptic curve utilized in the experiment is a MNT curve, with base field size of 159 bits and the embedding degree 6. The security level is chosen to be 80 bit, which means |_i| = 80 and |p| = 160. All experimental results represent the mean of 20 trials.

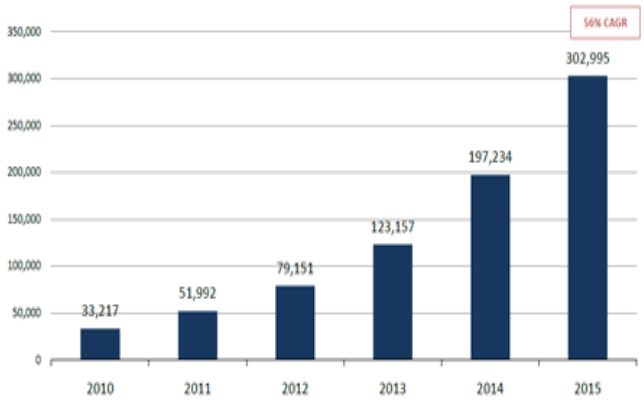


Fig.4 Cloud Storage Growth

According to [2], if t fraction of the file is corrupted, by asking proof for a constant c blocks of the file, the verifier can detect this server mis sbehavior with probability p = 1 − (1 − t)c. Let t = 1 − ρ and we get the variant of this relationship p = 1−ρc. Under this setting, we quantify the extra cost introduced by the support of dynamic data in our scheme into server computation, verifier computation as well as communication overhead.

V. RELATED WORKS

Information Support Systems (ISS) are computer technology/network support systems that interactively support the information processing mechanisms for individuals and groups in life, public, and private organizations, and other entities. Over some decades in the past, organizations have put efforts to be at the forefront of the development and application of computer-based Information Support Systems to collect, analyze and process the data and generate information to support decisions. Various computing paradigms have been employed for the purpose and needs have emerged for enormous infrastructure, unlimited system accessibility, cost effectiveness, increased storage, increased automation, flexibility, system mobility and shift of IT focus. This paper presents a brief evaluation on how Cloud Computing paradigm can be used to meet the increasing demands of the Information Support Systems and how Cloud Computing paradigm can prove to be future solution for such systems.

Using Cloud Storage, users can remotely store their data and enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance

Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third party auditor (TPA) to check the integrity of outsourced data and be worry-free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy, and introduce no additional online burden to user. In this paper, we propose a secure cloud storage system supporting privacy-preserving public auditing. We further extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient.

VI. IMPLEMENTATION

In this paper we have implemented PDP file-encoding functionality in order to test the effect of dispersal code choice on encoding time. The encryption process is needed while storing the data, and the data decryption is needed while retrieving the data. After the user's login has been successfully verified, if the CRM Service System requires client information from the user, it sends a request the information (for encryption and decryption) to the Storage Service System.

The code was written in C++ and experiments were run on an Intel Core 2 processor running at 2.16 GHz. All cryptographic operations utilize the RSA BSAFE C library. The dispersal code was implemented using the optimized library written in C. In order to implement the integrity protected, PRF values are added to the fragments stored on secondary servers. One subtle issue when implementing the IP-ECC construction is that the symbol size of Reed-Solomon encoding should be equal to the Security Parameter However, implements codes with symbol sizes up to 32 there are a number of interesting PDP variants to explore in follow-up work. The protocols we have described above for PDP only provide assurance for static files. We are investigating in current work design of similar protocols that accommodate file updates. We believe that the PDP techniques we have introduced in this paper help pave the way for valuable approaches to distributed file system availability

VII. CONCLUSION

In this paper, we tackle the privacy problem caused by the public auditing scheme. After presenting a new construction of *ASBB* scheme, we propose an efficient zero knowledge privacy preserving public auditing scheme for data storage security in cloud computing, i.e. the adversary cannot deduce any information of the file stored through the auditing interaction between CS and TPA. 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. Considering the time, computation resources, and even the related online burden of users, we also provide the

extension of the proposed main scheme to support third-party auditing, where users can safely delegate the integrity checking tasks to third-party auditors and be worry-free to use the cloud storage services. Through detailed security and extensive experiment results, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks. Future research should therefore be devoted to the design of an overall framework, integrating all the presented solutions, and activating the most appropriate solutions dependent on the current device, network and cloud server status.

REFERENCES

[1]. H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. of ASI- ACRYPT'08. Springer-Verlag, 2008, pp. 90–107.

[2]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. of CCS'07. New York, NY, USA: ACM, 2007, pp. 598–609.

[3]. A. Juels and B. S. Kaliski, Jr., "Pors: proofs of retrievability for large files," in Proc. of CCS'07. New York, NY, USA: ACM, 2007, pp. 584–597.

[4]. Amazon.com, "Amazon s3 availability event: July 20, 2008," Online at http:// status.aws.amazon.com/s3-20080720.html, July
2008.

[5]. J. Black, S. Halevi, H. Krawczyk, T. Krovetz, and P. Rogaway. UMAC: Fast and secure message authentication. In *CRYPTO*, volume 1666 of *LNCS*, pages 216–233, 1999.

[6]. K. D. Bowers, A. Juels, and A Oprea. HAIL: A High-availability and integrity layer for cloud storage, 2008. IACR ePrint manuscript 2008/489.

[7]. C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl.
Asynchronous verifiable secret sharing and proactive cryptosystems. In *9th ACM CCS*, pages 88–97, 2002.

[8.]C. Cachin and S. Tessaro. Asynchronous verifiable information dispersal. In *24th IEEE SRDS*, pages 191–202, 2005.

[9]. L. Carter and M. Wegman. Universal hash functions. *Journal of Computer and System Sciences*, 18(3), 1979.

[10]. R. Curtmola, O. Khan, and R. Burns. Robust remote data checking. In *4th ACM StorageSS*, 2008.

[11]. R. Curtmola, O. Khan, R. Burns, and G. Ateniese. MR-PDP:
Multiple-replica provable data possession. In *28th IEEE ICDCS*, pages 411–420, 2008.

[12]. K. D. Bowers, A. Jules, and A Oprea. Proofs of retrievability: Theory and implementation, 2008. IACR ePrint manuscript 2008/175.

[13]. A. Herzberg, M. Jakobsson, H. Krawczyk, and M. Yung. Proactive public key and signature systems. In *4th ACM CCS*, pages 100–110, 1997.

[14]. A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing, or: How to cope with perpetual leakage. In *CRYPTO*, volume 1963 of *LNCS*, pages 339–352, 1995.

[15]. A. Juels and B. Kaliski. PORs: Proofs of retrievability for large files. In *14th ACM CCS*, pages 584–597, 2007.

[16].H. Krawczyk. LFSR-based hashing and authentication. In *CRYPTO*, volume 839 of *LNCS*, pages 129–139, 1994.

[17].M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard. A cooperative Internet backup scheme. In *USENIX Annual Technical Conference*, pages 29–41, 2003.

[18].S. Micali, C. Peikert, M. Sudan, and D. Wilson. Optimal error correction against computationally bounded noise. In *TCC*, pages 1–16.

[19]. J. S. Plank and Y. Ding, "Note: Correction to the 1997 tutorial on reed-solomon coding," University of Tennessee, Tech. Rep. CS-03-
504, April 2003.

[20]. C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for storage security in cloud computing," in *Proc. of IEEE INFOCOM'10*, San Diego, CA, USA, March 2010.

[21]. S. Wilson, "Appengine outage," Online at http://www. cio-weblog.com/50226711/appengine outage.php, June 2008.

[22]. R. C. Merkle, "Protocols for public key cryptosystems," in *Proc. Of IEEE Symposium on Security and Privacy*, Los Alamitos, CA, USA,
1980.

[23]. Q. Wang, K. Ran, W. Lou, and Y. Zhang, "Dependable and secure sensor data storage with dynamic integrity assurance," in *Proc. Of IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Appril 2009.

[24].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.

[26]. M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *Proc. of Crypto'96, volume 1109 of LNCS*. Springer-Verlag, 1996, pp. 1–15.

[27]. T. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in *Proc. of ICDCS'06*, 2006, pp. 12–12.