IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

Integrity Constraints for Cloud Auditing Services Using Third Party Services

M. Anantha Lakshmi[1], Shaik Mahammad Rasheed[2]

Asst. Professor, Dept. of CSE, Ravindra College of Engineering for Women, Kurnool, Andhra Pradesh, India[1]

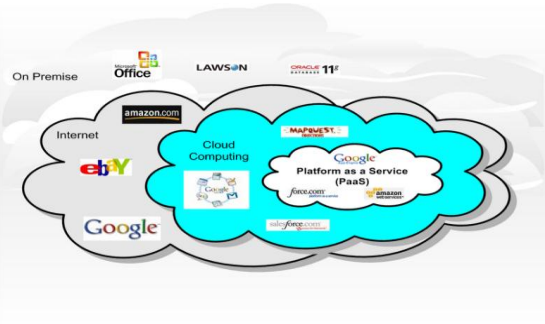
PG Scholar, Dept. of CSE, Mahatama Gandhi institute of technology, Hyderabad, Telangana State, India[2]

Abstract: Cloud data storage is the main important feature in present dynamic software cloud applications. Thus, enabling public audit ability 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. 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. We propose in this paper a flexible distributed storage integrity auditing mechanism, utilizing the homomorphic token and distributed erasure-coded data. The proposed 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 colluding attacks.

Keywords: Data integrity, dependable distributed storage, error localization, data dynamics, cryptographic protocols, cloud computing.

I. INTRODUCTION

Cloud computing is the delivery of computing services over the Internet. Cloud services allow individuals and businesses to use software and hardware that are managed by third parties at remote locations. Examples of cloud services include online file storage, social networking sites, webmail, and online business applications. The cloud computing model allows access to information and computer resources from anywhere that a network connection is available. Cloud computing provides a shared pool of resources, including data storage space, networks, computer processing power, and specialized corporate and user applications.



Cloud Computing has been envisioned as the next-generation information technology (IT) architecture for enterprises, due to its long list of unprecedented advantages in the IT history: on-demand self-service, ubiquitous network access, location independent resource pooling, rapid resource elasticity, usage-based pricing and transference of risk. 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 (CSP) for the availability and integrity of their data. On the one hand, although the cloud infrastructures are much more powerful and reliable than personal computing devices, broad range of both internal and external threats for data integrity still exist.

Privacy preserving approaches in cloud is 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 homomorphic token with distributed verification of erasure-coded data, our scheme achieves the storage correctness insurance as well as data error localization: whenever data corruption

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

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 order to save 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 (TPA) and be worry-free to use the cloud storage services.

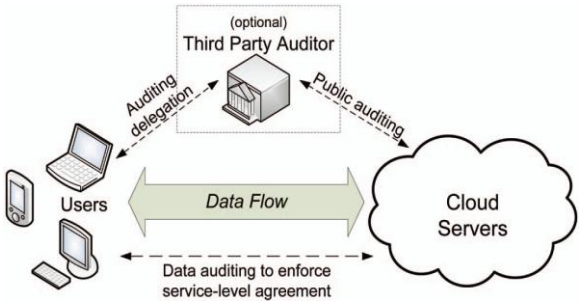


Figure 2: Cloud storage service architecture

Our work is among the first few ones in this field to consider distributed data storage security in cloud computing. Our contribution can be summarized as the following three aspects: 1) Compared to many of its predecessors, which only provide binary results about the storage status across the distributed servers, the proposed scheme achieves the integration of storage correctness insurance and data error localization, i.e., the identification of misbehaving server(s). 2) Unlike most prior works for ensuring remote data integrity, the new scheme further supports secure and efficient dynamic operations on data blocks, including: update, delete, and append. 3) 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.

II. RELATED WORK

We consider a cloud data storage service involving three different entities. Users rely on the CS for cloud data storage and maintenance. They may also dynamically interact with the CS to access and update their stored data for various application purposes. To save the computation resource as well as the online burden, cloud users may resort to TPA for ensuring the storage integrity of their outsourced data, while hoping to keep their data private from TPA. Representative network architecture for cloud storage service architecture is illustrated in Fig. 2. Three different network entities can be identified as follows: User: an entity, who has data to be stored in the cloud and relies on the cloud for data storage and computation, can be either enterprise or individual customers. Cloud Server (CS): an entity, which is managed by cloud service provider (CSP) to provide data storage service and has significant storage space and computation resources (we will not differentiate CS and CSP hereafter). . Third-Party Auditor: 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. As users no longer possess their data locally, it is of critical importance to ensure users that their data are being correctly stored and maintained. That is, users should be equipped with security means so that they can make continuous correctness assurance (to enforce cloud storage service-level agreement) of their stored data even without the existence of local copies.

Adversary Model

From user's perspective, the adversary model has to capture all kinds of threats toward his cloud data integrity. Because cloud data do not reside at user's local site but at CSP's address domain, these threats can come from two different sources: internal and external attacks. For internal attacks, a CSP can be self-interested, untrusted, and possibly malicious. Not only does it desire to move data that has not been or is rarely accessed to a lower tier of storage than agreed for monetary reasons, but it may also attempt to hide a data loss incident due to management errors, Byzantine failures, and so on. For external attacks, data integrity threats may come from outsiders who are beyond the control domain of CSP, for example, the economically motivated attackers. They may compromise a number of cloud data storage servers in different time intervals and subsequently be able to modify or delete users' data while remaining undetected by CSP.

III. ENSURING CLOUD DATA STORAGE

In cloud data storage system, users store their data in the cloud and no longer possess the data locally. Thus, the correctness and availability of the data files being stored on the distributed cloud servers must be guaranteed. 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, since it can always be the first step fast recover the storage errors or identifying potential threats of external attacks. To address these problems, our main scheme ensuring cloud data storage is presented in this section. The first part of the section is devoted to a review of basic tools from coding theory that is needed in our scheme for file distribution across cloud servers. Then, the homomorphic token is introduced.

File Distribution Preparation

It is well known that erasure-correcting code may be used to tolerate multiple failures in distributed storage systems. In cloud data storage, we rely on this technique to disperse the data file F redundantly across a set of n ¼ m þ k

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 4, April 2016

distributed servers. An ðm; kÞ Reed-Solomon erasure-correcting code is used to create k redundancy parity vectors from m data vectors in such a way that the original m data vectors can be reconstructed from any m out of the m þ k data and parity vectors. By placing each of the m þ k vectors on a different server, the original data file can survive the failure of any k of the m þ k servers without any data loss, with a space overhead of k=m. For support of efficient sequential I/O to the original file, our file layout is systematic, i.e., the unmodified m data file vectors together with k parity vectors is distributed across m þ k different servers.

IV. PROPOSED SCHEME

In order to achieve assurance of data storage correctness and data error localization simultaneously, our scheme entirely relies on the precomputed verification tokens. Upon receiving challenge, each cloud server computes a short "signature" over the specified blocks and returns them to the user. The values of these signatures should match the corresponding tokens precomputed by the user. Meanwhile, as all servers operate over the same subset of the indices, the requested response values for integrity check must also be a valid codeword determined by the secret matrix P.

Algorithm 1. Token Precomputation.
1: procedure
2: Choose parameters l; n and function f;_;
3: Choose the number t of tokens;
4: Choose the number r of indices per verification;
5: Generate master key KPRP and challenge key kchal;
6: for vector GðjÞ; j 1; n do
7: for round i 1; t do
8: Derive _i ¼ fkchal ðiÞ and kðiÞ prp from KPRP.
9: Compute vðjÞ i ¼ Pr
q¼1 _q
i _ GðjÞ½_kðiÞ prp ðqÞ_
10: end for
11: end for
12: Store all the vi's locally.
13: end procedure

Figure 3: Precipitation in token generation of cloud storage system

Error localization is a key prerequisite for eliminating errors in storage systems. It is also of critical importance to identify potential threats from external attacks.

A. Correctness Verification and Error Localization
Error localization is a key prerequisite for eliminating errors in storage systems. It is also of critical importance to identify potential threats from external attacks. However, many previous schemes do not explicitly consider the problem of data error localization, thus only providing binary results for the storage verification. Our scheme outperforms those by integrating the correctness verification and error localization (misbehaving server identification) in our challenge-response protocol: the response values from servers for each challenge not only determine the correctness of the distributed storage, but also contain information to locate potential data error(s).

Algorithm 2. Correctness Verification and Error Localization.
1: procedure CHALLENGE(i)
2: Recompute _i ¼ fkchal ðiÞ and kðiÞ prp from KPRP ;
3: Send f_i; kðiÞ prpg to all the cloud servers;
4: Receive from servers:
fRðjÞ i ¼ Pr
q¼1 _q
i _ GðjÞ½_kðiÞ prp ðqÞ_j1 _ j _ ng
5: for ðj m þ 1; nÞ do
6: RðjÞ RðjÞ _Pr
q¼1 fkj ðsIq ;jÞ _ _q
i , Iq ¼ _kðiÞ prp ðqÞ
7: end for
8: if ððRð1Þ i ; . . .;RðmÞ i Þ _ P¼¼ðRðmþ1Þ i ; . . .;RðnÞ i ÞÞ than
9: Accept and ready for the next challenge.
10: else
11: for (j 1; n) do
12: if ðRðjÞ i ! ¼vðjÞ i Þ than
13: return server j is misbehaving.
14: end if
15: end for
16: end if
17: end procedure

B. File Retrieval and Error Recovery
Since our layout of file matrix is systematic, the user can reconstruct the original file by downloading the data vectors from the first m servers, assuming that they return the correct response values. Notice that our verification scheme is based on random spot-checking, so the storage correctness assurance is a probabilistic one. However, by choosing system parameters ðe:g:; r; l; tÞ appropriately and conducting enough times of verification, we can guarantee the successful file retrieval with high probability. On the other hand, whenever the data corruption is detected, the comparison of precomputed tokens and received response values can guarantee the identification of misbehaving server(s) (again with high probability), which will be discussed shortly. Therefore, the user can always ask servers to send back blocks of the r rows specified in the challenge and regenerate the correct blocks by erasure correction, shown in Algorithm 3, as long as the number of identified misbehaving servers is less than k. (otherwise, there is no way to recover the corrupted blocks due to lack of redundancy, even if we know the position of misbehaving servers.) The newly recovered blocks can then be redistributed misbehaving servers to maintain the correctness of storage.

Algorithm 3. Error Recovery.
1: procedure
% Assume the block corruptions have been detected among
% the specified r rows;
% Assume s _ k servers have been identified misbehaving
2: Download r rows of blocks from servers;

3: Treat s servers as erasures and recover the blocks.
4: Resend the recovered blocks to corresponding servers.
5: end procedure

V. EXPERIMENTAL RESULTS

We now assess the performance of the proposed storage auditing scheme. We focus on the cost of file distribution preparation as well as the token generation. Our experiment is conducted on a system with an Intel Core 2 processor running at 1.86 GHz, 2,048 MB of RAM, and a 7,200 RPM Western Digital 250 GB Serial ATA drive.

File Distribution Preparation

File distribution preparation includes the generation of parity vectors (the encoding part) as well as the corresponding parity blinding part. We consider two sets of different parameters for the ðm; kÞ Reed-Solomon encoding, both of which work over GFð216Þ. This can be explained as follows: on the one hand, k determines how many parity vectors are required before data outsourcing, and the parity generation cost increases almost linearly with the growth of k; on the other hand, the growth of k means the larger number of parity blocks required to be blinded, which directly leads to more calls to our non optimized PRF generation in C.

Challenge Token Computation

Although in our scheme the number of verification token t is a fixed priori determined before file distribution, we can overcome this issue by choosing sufficient large t in practice. For example, when t is selected to be 7,300 and 14,600, the data file can be verified every day for the next 20 years and 40 years, respectively, which should be of enough use in practice. Following the security analysis, we select a practical parameter r ¼ 460 for our token precomputation (see Section 5.2.1), i.e., each token covers 460 different indices. Other parameters are along with the file distribution preparation. Our implementation shows that the average token precomputation cost is about 0.4ms. This is significantly faster than the hash function based token precomputation scheme proposed in [14]. To verify encoded data distributed over a typical number of 14 servers, the total cost for token precomputation is no more than 1 and 1.5 minutes, for the next 20 years and 40 years, respectively. Note that each token is only an element of field GFð216Þ, the extra storage for those precomputed tokens is less than 1MB, and thus can be neglected.

VI. CONCLUSION

Consider the process data cloud storage using third party service provider. We utilize the homomorphic linear authenticator and random masking to guarantee that the TPA would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage. The proposed 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.

REFERENCES

[1] Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring Data Storage Security in Cloud Computing," Proc. 17th Int'l Workshop Quality of Service (IWQoS '09), pp. 1-9, July 2009.
[2] Amazon.com, "Amazon Web Services (AWS)," http://aws. amazon.com, 2009.
[3] M. Arrington, "Gmail disaster: Reports of mass email deletions," Online at http://www.techcrunch.com/2006/ 12/28/gmail-disasterreports-of-mass-email-deletions/, December 2006.
[4] J. Kincaid, "MediaMax/TheLinkup Closes Its Doors," Online at ttp://www.techcrunch.com/2008/07/10/ mediamax thelinkup-closes-its-doors/, July 2008.
[5] Amazon.com, "Amazon s3 availability event: July 20, 2008," Online at http://status.aws.amazon.com/s3-20080720.html,2008.
[6] S. Wilson, "Appengine outage," Online at http://www. cio-weblog.com/50226711/appengine outage.php, June 2008.
[7] B. Krebs,"Payment Processor Breach May Be Largest Ever," Online at http://voices.washingtonpost.com/securityfix/2009/01/ payment processor breach may b.html,Jan 2009.
[8] 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, Alexandria, VA, October 2007, pp. 598–609.
[9] B. Krebs, "Payment Processor Breach May Be Largest Ever," http://voices.washingtonpost.com/securityfix/2009/01/ payment_processor_breach_may_b.html, Jan. 2009.
[10] 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, Oct. 2007.

BIOGRAPHIES

Mrs. M. Anantha Lakshmi has completed B.Tech in Computer Science and Information Technology from G. Pulla Reddy Engineering College (Autonomous), affiliated to SKU, Kurnool, in the year 2012 and Completed her M. Tech from G. Pulla Reddy Engineering College (Autonomous), affiliated to JNTUA, Kurnool, in the year 2014. Presently she is working as Asst. Professor in the Department of Computer Science & Engineering at Ravindra College of Engineering for Women, Kurnool. She has presented two international journals so far her research areas include Cloud Computing.

Mr. Shaik Mahammad Rasheed obtained his B.Tech degree from Kottam College of Engineering, Kurnool and M.Tech degree from Mahatama Gandhi Institute of Technology, Hyderabad in the year 2012 and 2014 respectively. He has presented two international journals so far his research areas include Computer Networks and Network Security.