RESEARCH ARTICLE OPEN ACCESS

Reliable Storage Services and Avoidance of Integrity Problems in Cloud Computing

T.Subetha[1], M.Rajasekar[2]
[1]ME CSE
Vel Tech Multi Tech Engineering College, Avadi, Chennai
[1]subethathankaraj@gmail.com
[2]Asst Professor, Dept of Computer Science and Engineering,
Vel Tech Multi Tech Engineering College, Avadi, Chennai.
[2]mrajasekarcse@gmail.com

Abstract—
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. Although the cloud infrastructures are much more powerful and reliable than personal computing devices, broad range of internal and external threats for data integrity still exist. The users may not retain a local copy of outsourced data; there exist various incentives for cloud service providers (CSP) to behave unfaithfully towards the cloud users regarding the status of their outsourced data, which leads to storage incorrectness. In order to address this new problem, we propose in this paper a flexible distributed storage integrity auditing mechanism, utilizing the homomorphic token and distributed erasure-coded data. The auditing result ensures strong cloud storage correctness and avoids the integrity problems

Key Terms—*Data integrity, dependable distributed storage, error localization, data dynamics, Cloud Computing*

I. INTRODUCTION

Several trends are opening up the era of Cloud Computing, which is an Internet-based 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 centers 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) [2] 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 [3]. 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. Examples of outages and data loss incidents of noteworthy cloud storage services appear from time to time [4]–[8]. On the other hand, since users may not retain a local copy of outsourced data, there exist various incentives for cloud service providers (CSP) to behave unfaithfully towards the cloud users regarding the status of their outsourced data. For example, to increase the profit margin by reducing cost, it is possible for CSP to discard rarely accessed data without being detected in a timely fashion [9]. Similarly, CSP may even attempt to hide data loss incidents so as to maintain a reputation [10]–[12]. Therefore, although outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, its lacking of offering strong assurance of data integrity and availability may impede its wide adoption by both enterprise and individual cloud users. In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. However, the fact that users no longer have physical possession of data in the cloud prohibits the direct adoption of traditional

cryptographic primitives for the purpose of data integrity protection. Hence, the verification of cloud storage correctness must be conducted without explicit knowledge of the whole data files [9]–[12]. Meanwhile, cloud storage is not just a third party data warehouse. The data stored in the cloud may not only be accessed but also be frequently updated by the users [13]–[15], including insertion, deletion, modification, appending, etc. Thus, it is also imperative to support the integration of this dynamic feature into the 2 cloud storage correctness assurance, which makes the system design even more challenging. Last but not the least, the deployment of Cloud Computing is powered by data centers running in a simultaneous, cooperated and distributed manner [3]. It is more advantages for individual users to store their data redundantly across multiple physical servers so as to reduce the data integrity and availability threats. Thus, distributed protocols for storage correctness assurance will be of most importance in achieving robust and secure cloud storage systems. However, such important area remains to be fully explored in the literature.

Recently, the importance of ensuring the remote data integrity has been highlighted by the following research works under different system and security models [9] [19]. These techniques, while can be useful to ensure the storage correctness without having users possessing local data, are all focusing on single server scenario. They may be useful for quality-of-service testing [20], but does not guarantee the data availability in case of server failures. Although direct applying these techniques to distributed storage (multiple servers) could be straightforward, the resulted storage verification overhead would be linear to the number of servers. As an complementary approach, researchers have also proposed distributed protocols [20]–[22] for ensuring storage correctness across multiple servers or peers. However, while providing efficient cross server storage verification and data availability insurance, these schemes are all focusing on static or archival data. As a result, their capabilities of handling dynamic data remains unclear, which inevitably limits their full applicability in cloud storage scenarios.

In this paper, 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 [23], 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 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 and be orry-free to use the cloud storage services. 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. The rest of the paper is organized as follows. Section II introduces the system model, adversary model, our design goal and notations. Then we provide the detailed description of our scheme in Section III and IV. Section V gives the security analysis and performance evaluations, followed by Section VI which overviews the related work. Finally, Section VII concludes the whole paper.

II. PROBLEM STATEMENT

2.1 System Model

A representative network architecture for cloud storage service architecture is illustrated in Figure 1. Three different network entities can be identified as follows:

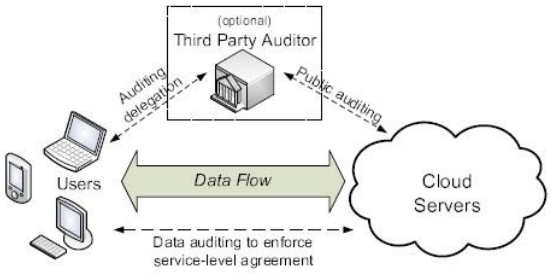
• User: an entity, who has data to be stored in the cloud and relies on the cloud for data storage and computation, can be either enterprise or individual

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 (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. In cloud data storage, a user stores his data through a CSP into a set of cloud servers, which are running in a simultaneous, cooperated and distributed manner. Data redundancy can be employed with technique of erasure correcting code to further tolerate faults or server crash as user's data grows in size and importance. Thereafter, for application purposes, the user interacts with the 3 cloud servers via CSP to access or retrieve his data. In some cases, the user may need to perform block level operations on his data. The most general forms of these operations we are considering are block update, delete, insert and append. Note that in this paper, we put more focus on the support of file-oriented cloud applications other than non-file application data, such as social networking data. In other words, the cloud data we are considering is not expected to be rapidly changing in a relative short period. 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. In case that users do not necessarily have the time, feasibility or resources to monitor their data online, they can delegate the data auditing tasks to an optional trusted TPA of their respective choices. However, to securely introduce such a TPA, any possible leakage of user's outsourced data towards TPA through the auditing protocol should be prohibited. In our model, we assume that the point-to-point communication channels between each cloud server and the user is authenticated and reliable, which can be achieved in practice with little overhead. These authentication handshakes are omitted in the following presentation.



Cloud storage service architecture

2.2 Adversary Model

From user's perspective, the adversary model has to capture all kinds of threats towards 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. Therefore, we consider the adversary in our model has the following capabilities, which captures both external and internal threats towards the cloud data integrity. Specifically, the adversary is interested in continuously corrupting the user's data files stored on individual servers. Once a server is comprised, an adversary can pollute the original data files by modifying or introducing its own fraudulent data to prevent the original data

2.3 Design Goals

To ensure the security and dependability for cloud data storage under the aforementioned adversary model, we aim to design efficient mechanisms for dynamic data verification and operation and achieve the following goals: (1) Storage correctness: to ensure users that their data are indeed stored appropriately and kept intact all the time in the cloud. (2) Fast localization of data error: to effectively locate the malfunctioning server when data corruption has been detected. (3) Dynamic data support: to maintain the same level of storage correctness assurance even if users modify, delete or append their data files in the cloud. (4) Dependability: to enhance data availability against Byzantine failures, malicious data modification and

server colluding attacks, i.e. minimizing the effect brought by data errors or server failures. (5) Lightweight: to enable users to perform storage correctness checks with minimum overhead.

2.4 PROVIDING DYNAMIC DATA OPERATION SUPPORT

So far, we assumed that **F** represents static or archived data. This model may fit some application scenarios, such as libraries and scientific datasets. However, in cloud data storage, there are many potential scenarios where data stored in the cloud is dynamic, like electronic documents, photos, or log files etc. Therefore, it is crucial to consider the dynamic case, where a user may wish to perform various block-level operations of update, delete and append to modify the data file while maintaining the storage correctness assurance.bSince data do not reside at users' local site but atbcloud service provider's address domain, supporting dynamic data operation can be quite challenging. On the one hand, CSP needs to process the data dynamics request without knowing the secret keying material. On the other hand, users need to ensure that all the dynamic data operation request has been faithfully processed by CSP. To address this problem, we briefly explain our approach methodology here and provide the details later. For any data dynamic operation, the user must first generate the corresponding resulted file blocks and parities. This part of operation has to be carried out by the user, since only he knows the secret matrix P. Besides, to ensure the changes of data blocks correctly reflected in the cloud address domain, the user also needs to modify the corresponding storage verification tokens to accommodate the changes on data blocks. Only with the accordingly changed storage verification tokens, the previously discussed challenge-response protocol can be carried on successfully even after data dynamics. In other words, these verification tokens help ensure that CSP would correctly execute the processing of any dynamic data operation request. Otherwise, CSP would be caught cheating with high probability in the protocol execution later on. Given this design methodology, the straightforward and trivial way to support these operations is for user to download all the data from the cloud servers and re-compute the whole parity blocks as well as verification tokens. This would clearly be highly inefficient. In this section, we will show how our scheme can explicitly and efficiently handle dynamic data operations for cloud data storage, by utilizing the linear property of Reed-Solomon code and verification token construction.

III. CONCLUSION

In this paper, we investigate the problem of data security in cloud data storage, which is essentially a distributed storage system. To achieve the assurances of cloud data integrity and availability and enforce the quality of dependable cloud storage service for users, we propose 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). 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.

REFERENCES

[1] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in *Proc. of IWQoS'09*, July 2009, pp. 1–9.

[2] Amazon.com, "Amazon web services (aws)," Online at http:// aws.amazon.com/, 2009.

[3] Sun Microsystems, Inc., "Building customer trust in cloud computing with transparent security," Online at https://www.sun. com/offers/details/sun transparency.xml, November 2009.

[4] M. Arrington, "Gmail disaster: Reports of mass email deletions," Online at http://www.techcrunch.com/2006/12/ 28/gmail-disasterreports-of-mass-email- eletions/, December 2006.

[5] J. Kincaid, "MediaMax/TheLinkup Closes Its Doors," Online at http://www.techcrunch.com/2008/07/10/ mediamaxthelinkup-closes-its-doors/, July 2008.

[6] Amazon.com, "Amazon s3 availability event: July 20, 2008," Online at http://status.aws.amazon.com/s3-080720.html, July 2008.

[7] S. Wilson, "Appengine outage," Online athttp://www.cioweblog.com/50226711/appengi ne outage.php, June 2008.

[8] B. Krebs, "Payment Processor Breach May Be Largest Ever," Online at http://voices.washingtonpost.com/securityfix/20 09/01/ payment processor breach may b.html, Jan. 2009.

[9] A. Juels and J. Burton S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. of CCS'07*, Alexandria, VA, October 2007, pp.584–597.