Research Paper Engineering

Ensuring Data Possession and Volatile Data Integrity in Cloud Computing

* S.Sharavanan * R.Vijai

* Professor, Annapoorana Engineering College, Salem

** Assistant Professor, Annapoorana Engineering College, Salem

ABSTRACT

Data storage in cloud involves user for remote storage of their data without the need for local software and hardware management. On he other hand the retrieval of user dala may result in incorrectness. To ensure the correctness of stored data in cloud, an independent auditing mechanism is proposed in this paper with low computation and communication cost. By Distributed erasure coded data mechanisnil3l and Homomorphic token, correctness of the stored data is ensured. Identification of malfunctioning servers and retrieving the original data from hidden servers is also achieved in case of server collusion attacks. The proposed design also supports the dyuaniicity over the outsourced data including block modification, deletion and append. The proposed scheme is capable of handling Byzantine famlurel7j and malicious attacks.

Keywords : Distributed erasure coded data, Hidden servers, Byzantine failures

1. INTRODUCTION

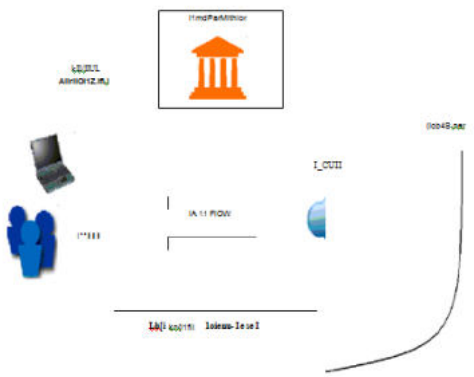
Cloud computing is the upcoming architecture in IT enterprise due to its vast advantages both in IT and Non-IT on demand self-service location-independent resource pooling, rapid resource elasticity, usage-based pricing[lj. cloud represents the 'internet' [providing services to users through internet managed by the cloud service providers (CSP)]. One firndamental aspect of this new computing model is that data is being centralized or outsourced into the cloud. The cloud is a means by which global class, highly scalable and flexible services can be delivered and consumed over the internet through an as-needed, pay-per-use business model[21. Multiple clients can share a common technology platform and even a single application instance. Though the cloud has large benefits over storage of data it also suffers from security concerns due to byzantine failures[31 and server colluding attacks. Hence, we require verification of data storage in the cloud. This is built in the proposed scheme by data auditing mechanism and homomorphic tokens. Cloud Computing is not just a third party data warehouse. The stored data in cloud may be frequently revised by the users, including operations like insertion deletion, modification, affixing, reordering, etc. To ensure storage correctness under dynamic data revise is hence of paramount importance. However, this dynamic feature also makes traditional integrity insurance techniques futile and entails new solutions. Individual user's data is redundantly stored in multiple physical locations to further reduce the data integrity threats

II. PROBLEM STATEMENT

A. System Model:

Representative network architecture for cloud data storage is shown below. Three different network entities can be identified as follows:

- User: users, who have data to be stored in the cloud and rely on the cloud for data computation, consist of both individual clients and organizations.
- Cloud Service Provider (CSP): a CSP, who has considerable resources and proficiency in building and managing dispersed cloud storage servers, owns and operates live Cloud Computing systems.
- Third Party Auditor (TPA): an not obligatory TPA, who has proficiency and capabilities that users may not have, is trusted to review and expose risk of cloud storage services on behalf of the users upon demand .In cloud data storage, a user stores his data through a CSP into a set of cloud servers, wffich are running in a synchronized, cooperated and disseminated approach. Data redundancy can be engaged with practice of erasure-correcting code[4] to auxiliary tolerate faults or server crash as user's data grows in size and significance. Thereafter for application purposes, the user interacts with the cloud servers via CSP to access or recover his data. In some cases the user may need to execute block level operations on his data The most common forms of these operations we are considering are block update, delete, insert and append As users no longer acquire their data in the vicinity, it is of significant importance to guarantee users that their data are being appropriately stored and maintained. That is, users should be outfitted with defense means so that they can make iicessant precision assurance of their stored data even without the subsistence of local copies [5]. In case that users do not inevitably have the time, viability or resources to scrutinize their data, they can entrust the tasks to an optional trusted TPA of their relevant choices.



III. ENABLING DATA STORAGE INTEGRITY

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

disseminated cloud servers must be assured. One of the key ssues is to effectively detect any unconstitutional data variation and corruption, possibly due to server compromise and/ or random Intricate failures. Besides, in the dispersed case when such inconsistencies are effectively detected, to find which server the data error lies in is also of great significance, since it can be the first step to fast detect the storage errors. The first part involves usage of homomorphic tokens to store the user data on distributed servers in cloud. The token

computation function we are considering belongs to a family of universal hash function[8], chosen to preserve the homomorphic properties, which can be perfectly integrated with the verification of erasure-coded data [6j.Subsequently, it is also shown how to derive a challenge response protocol for verifying the storage accuracy as well as identifying misbehaving servers.

A Homomorphic token computation mechanis,n:

IA 1.1 FIOW
indices. After getting assurance of the user it again asks for authentication by which the user is confirmed to be the authenticated user. Upon receiving assurance, 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 pre-computed 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 a secret matrix

Suppose the user wants to challenge the cloud server's times to make sure the conectness of data storage. Then, he must pre-compute 't' verification tokens for each function a challenge key and a master key are used. To generate the ith token for server j, the user acts as follows:

1. Derive a arbitrary value i and a permutation key based on master permutation key.
II. Compute the set of randomly-chosen indices.
III. Calculate the token using encoded file and the arbitrary
l "III

Lh[i kn{l1fll 1oienu-lesel value derived.

B. Data Integrity and Error Localization:
Existing scheme provides only binary results for the storage verification. Our scheme provides those by integrating the correctness verification and error localization 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).

Specifically, the procedure of the ith challenge-response for a cross-check over the n servers is described as follows:

i) The user reveals the i as well as the ith key k (i) to each servers
ii) The server storing vector 0 aggregates those r rows iii) Specified by index k(i) into a linear combination R
iv) Upon receiving R is from all the servers, the user takes away values in R.
v) Then the user verifies whether the received values remain a valid codeword determined by secret matrix.

Because all the servers operate over the same subset of indices, the linear aggregation of these r specified rows

(R(l)i R(n)i) has to be a codeword in the encoded file matrix. If the above equation holds, the challenge is passed Otherwise, it indicates that among those specified rows, there exist file block corruptions. Once the inconsistency among the storage has been successfully detected, we can rely on the pre computed verification tokens to ftirther determine where the potential data error(s) lies in. Note that each response R(j) its In order to achieve assurance of data storage correctness computed exactly in the same way as token vU) i , thus the user and data error localization, our scheme entirely relies on the pre computed verification tokens. The main idea is before file distribution the user pre-computes a certain number of short verification tokens on individual; each token covers a random subset of data blocks. Later, when the user wants to make sure the storage correctness for the data in the cloud, he challenges the cloud servers with a set of randomly generated block can simply find which server is misbehaving by verification

C. 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(e.g., r, l, t) 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 pre-computed tokens and received response values can guarantee the identification of misbehaving server(s), again with high probability. Therefore, the user can always ask servers to send back blocks of the r rows specified in the challenge and reinforce the correct blocks by erasure correction, as long as there are at most k misbehaving servers are acknowledged. The newly recovered blocks can then be re dispersed to the misbehaving servers to maintain the correctness of storage[5j.

IV. INTRODUCING THIRD PARTY AUDITOR
TPA is the trusted entity that has expertise and capabilities to assess cloud storage security on behalf of a data owner upon request. Under the cloud paradigm, the data owner may represent either the individual or the enterprise customer, who relies on the cloud server for remote data storage and maintenance, and thus is relieved of the burden of building and maintaining local storage infrastructure. In most cases cloud data storage services also provide benefits like availability (being able to access data from anywhere), relative low cost (paying as a function of need), and on demand sharing among a group of trusted users, such as partners in a collaboration team or employees in the enterprise organization. Only the data owner can dynamically interact with the CS to update her stored data, while users just have the privilege on file reading. Within the scope of this article, we focus on how to ensure publicly auditable secure cloud data storage services. As the data owner no longer possesses physical control of the data, it is of critical importance to allow the data owner to verify that his data is being correctly stored and maintained in the cloud. Considering the possibly large cost in terms of resources and expertise, the data owner may resort to a TPA for the data auditing task to ensure the storage security of her data, while hoping to keep the data private from the TPA. We assume the TPA, who is in the business of auditing, is reliable and independent, and thus has no incentive to collude with either the CS or the owners during the auditing process[7]. The TPA should be able to efficiently audit the cloud data storage without local copy of data and without any additional online burden for data owners. Besides, any possible leakage of an owner's outsourced data toward a TPA through the auditing protocol should be prohibited.

We consider both malicious outsiders and a semi-trusted CS as potential adversaries interrupting cloud data storage services. Malicious outsiders can be economically motivated, and have the capability to attack cloud storage servers and subsequently pollute or delete owners' data while remaining undetected. The CS is semi-trusted in the sense that most of the time it behaves properly and does not deviate from the prescribed protocol execution. However, for its own benefit tht CS might neglect to keep or deliberately delete rarely acccsse

data files that belong to ordinary cloud owners. Moreover, tht CS may decide to hide the data corruptions caused by servci hacks or Byzantine failures[7j.Some of the desirable properties of TPA are protecting data privacy which includes that ii should be able to efficiently audit the cloud data storagc without demanding a local copy of data or even learning tht data content and When receiving multiple auditing tasks fron different owners delegations, a TPA should still be able tc handle them in a fast yet cost-effective manner.

V. DYNAMIC DATA OPERATIONS

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 modification, delete and append to the data file while maintaining the storage correctness assurance.

The straightforward and insignificant 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 An effective way is proposed in this scheme.

A. Modifj Operation

In cloud data storage, sometimes the user may need to modify some data block(s) stored in the cloud, from its current value f to a new one. We refer to this operation as data modification.

B. Delete Operation

Sometimes, after being stored in the cloud, certain date blocks may need to be deleted. The delete operation we ar considering is a general one, in which user replaces the dart block with zero or some special reserved data symbol. Froir this point of view, the delete operation is actually a special casi of the data modify operation, where the original data blocks can be replaced with zeros or some predetermined special blocks.

C. Append Operation

In some cases, the user may want to increase the size of hn stored data by adding blocks at the end of the data file, whici we refer as data append. We anticipate that the most frequeni append operation in cloud data storage is bulk append, r which the user needs to upload a large number of blocks (not i single block) at one time.

VICONCLUSION

In this paper, we investigated the problem of data security in cloud data storage, which is fundamentally a dispersed storage system. To ensure the correctness of users' data in cloud data storage, we proposed an effective and flexible dispersed scheme with explicit dynamic data sustain, including block update, delete, and append. We rely on erasure- correcting code[3] in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. By utilizing the Homomorphic token with dispersed 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 dispersed servers, we can almost guarantee the simultaneous identification of the misbehaving server(s). We believe that data storage security in Cloud Computing, an area full of challenges and of dominant significance, is still in its infancy to be identified. It allows the Third party auditing system to ensure the data privacy with cost-effectiveness.

REFERENCES

[1j Peter Mell, Timothy Grance The NIST Definition of Cloud Computing", September 2011 Amazon web services: http://aws.amazon.comlec2/, 2012.

[2] Mahul A. Shah, Mary Baker, Jeffrey C. Mogul, Ram Swaminathan, "Auditing to Keep Online Storage Services Honest", 2007.

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

[4j M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," ACM Transaction on Computer Systems,vol20, Crypto'96, volume 1109 of

[5j C.Wang,K.Ren,W.Lou, and J.Li,"Towards Publicly auditable secure cloud data storage services," IEEE Network Magazine, vol. 24, no. 4, pp. 19—24, 2010.

[6] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in Proc. of LNCS. Springer-Verlag, 1996, pp. 1–15.