Research Paper Engineering

Ensuring Availability and Integrity of Data Storage in Cloud Computing

*K. Sunitha ** V. Tejaswini ***S.K. Prashanth

*,** M.Tech (C.S.E) VCE (AICTE), Hyderabad, India

*** Prof. CSE, VCE (AICTE), Hyderabad India

ABSTRACT

Cloud storage is a cloud computing model in which data is stored on remote servers accessed from the Internet. Eventhough there is a benefit of accessing the data from the remote servers there raises a security problem. In order to overcome the problem, we proposed a efficient and effective verification scheme to ensure the integrity and availability of the data using homomorphic token and distributed erasure coded data. Considering data are dynamic in nature, then it further supports dynamic operations including such as modify, update, insert and append. This paper highlights the Fermat Number Transform based Reed-Solomon erasure code to achieve the availability of data and drastically reduce the computation time and overhead.

Keywords: Data integrity, Data availability, Cloud computing, Data dynamics

1. Introduction

Cloud computing is an Internet-based computing and use of computer technology which provides access resources as a service such as storage, network, server and processors etc. By moving data into the cloud offers great ease to users. In general data center in cloud holds information that users have store on their computers. The security concerns arises because the outsource data is used by the user.

In order to achieve the security assurance of cloud data availability and integrity efficient and effective methods are used. Still, the fact those users no longer have local copy of data in the cloud prohibits the direct adoption of traditional cryptographic primitives for the purpose of data integrity protection. For this reason, the verification of cloud storage integrity must be conducted without explicit knowledge of the whole data files [3], [4], [5]. We considering, cloud data is dynamic in data storage i.e. the data stored in the cloud supports dynamic operations like insertion, deletion, appending and modification. This dynamic feature into the cloud gives the storage correctness assurance.

Ensuring of remote data integrity has been highlighted by different system and security models. Eventhough, after applying these techniques to multiple servers could be straight forward and all are focusing on static data.

Recently, Wang et al [16] proposed a Toward Secure and Dependable Storage Services in Cloud Computing to ensure the availability and integrity of outsourced data in cloud using reed-solomon code. However, their scheme is not more efficient, because they are using Vandermonde Reed-Solomon erasure code for data availability. This code takes long time for encoding the file, which is inefficient.

In this paper, we propose an effective and efficient distributed storage verification scheme with explicit dynamic data support to ensure the integrity and availability of users outsourced data in the cloud server. We rely on Fermat Number Transform (FNT) based Reed-Solomon erasure code in the file distribution preparation instead of Vandermonde Reed-Solomon code to provide redundancies and guarantee the data availability against Byzantine servers. This construction significantly reduces the computation time and storage overhead as compared to the traditional distribution techniques.

2. Problem Formulation and System Model

2.1 System Model

The cloud storage service architecture is illustrated in network representations as follows in Fig. 1. Three different network entities can be recognized as follows:

User: Who have data to be stored in the cloud and rely on the cloud for data storage computation, can be either enterprise or individual customers.

Cloud Server (CS): an entity, which is managed by cloud service provider (CSP), who has significant resources and building and managing distributed cloud storage servers.

Third-Party Auditor: TPA has an optional, who has expertise and capabilities that users do not have and is trusted to assess the cloud storage security and behalf of the users upon request.

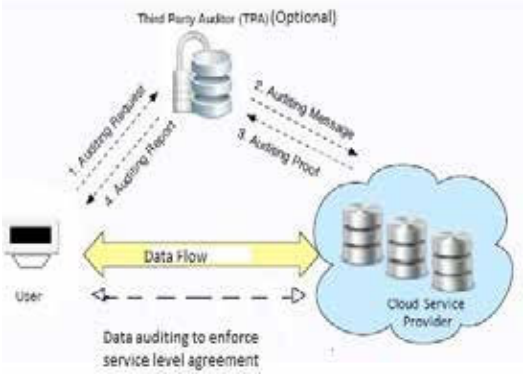

Figure 1 Cloud computing basic architecture

In the cloud storage, a user stores his data into a set of cloud servers using CSP, which are running in a Fig.1. The user interacts with the cloud servers via CSP to access or retrieve his data. But in some cases, the user may need to perform block level operations on his data. In case that users do not

have the time, they can delegate the data auditing tasks to an optional trusted TPA choices.

In order to ensure the security and dependability of the Cloud Server a special entity is used, which is referred to as an adversary model. The adversary is interested in continuously corrupting the users data which is stored on individual servers. Once a server is comprised,an adversary can corrupt the original data files by modify or introducing its own fraudulent data in order to prevent the original data from being retrieved by the user.

2.2 Design Goals
The main goals for integrity and of this paper are;

1. The challenge-response protocol will provide the localization of data error.
2. We propose an efficient method for encoding the data file to be transferred and stored in the Cloud.
3. Finally, we propose an efficient data recovery method in order to retrieval the of lost data in Cloud.
4. Dynamic data support which is to maintain the same level of storage correctness assurance even if users modify, insert, delete or append their data files in the cloud.

2.3 Preliminaries and notations

- D-the data file to be stored in cloud, we assume that D is denoted as matrix of 'm' equal sized data blocks each consisting of '*l*' data blocks, these all data blocks belongs Galois Field GF (2^w) where w=8 or 16.
- A-the generator distribution matrix used for Fermat Reed-Solomon encoding.
- G-is the encoded file matrix, which contains both data and parity blocks, where each consisting of '*l*' blocks.
- f_{key}(.)-pseudo Random Function (PRF) indexed on some *key*, which is defined as $f : \{0,1\}^* \times key\text{-GF}(2^w)$.
- $\pi_{key(.)}$-pseudo Random Permutation (PRP) indexed under *key,* which is defined as $\pi : \{0,1\}^{\log 2(l)} \times key - \{0,1\}^{\log 2(l)}$.

3. Proposed Protocol
To provide availability and integrity of data storage in cloud computing, we propose a new data verification protocol. It is designed based on Fermat number transform and spot checking using hash values. Our protocol consists of four phases: 1) File Encoding 2) Token Computation 3) Data Integrity Checking and Dynamic Data Operations.

3.1 File Encoding
To attain availability of data stored in cloud we are using FNT based Reed-Solomon erasure code. Generally erasures are used to recover the data loss in distributed storage systems. This is based on FNT support practical encoding and decoding algorithms with complexity $O(n \log n)$, where n is the number of symbols of a codeword. Particulary, using this approach we are benified by the connection of erasure codes with FNT allow to reuse the optimized software and hardware implementations developed for the applications.

Assume that q is a prime number, the values 0,1…….,q<1 form a finite field where addition and product are processed modulo q. This finite field is known as the Galois field GF(q). In finite fields the order of a number is defined as the lowest power of the number which is equals 1 modulo q. An element of the Galois field is called a primitive root of the field if its order is q-1. Consider example, 3 is a primitive root in Galois field GF(65537), because $3^{65536} \equiv 1 \bmod 65537$ and for each 0<i<65536, $3^i \neq 1 \bmod 65537$.

The principles of the Fourier transform can be extensive to these finite fields, as introduced by Pollard[ ]. Let r be an element of order n-1 in the field. In this case, the Discrete Fourier transform (DFT), which take a vector a = ($a_0, a_1, \ldots, a_{n-1}$) of size n as input in GF(q), returns

$$A_j = \sum_{i=0}^{} a_i r^j \quad , 0 \le j \le n-1, n \le q$$

where A is a vector of size n. In the same way, the inverse DFT (DFT^{-1}) can be defined as:

$$a_j = \frac{1}{n} \times \sum_{i=0}^{n-1} A_i r^{-j} \quad ,0 \le j \le n-1, n < q.$$

In order to process the DFT, Fermat Number Transform is used in finite fields. FNT is the equivalent of the FFT algorithm on Fermat fields [10]. By using divide and conquer approach, the complexity of the FNT of size n and its inverse FNT^{-1}, reduces the process of the DFT to O(n log n). The FNT can also be represented by a n × n square matrix. This matrix is a special case of a Vandermonde matrix on special set $1, r, r^2, \ldots\ldots, r^{n-1}$, where r is of order n-1. Since these values are pairwise dissimilar, the matrix is invertible. The first k rows of this FNT matrix form the generator matrix of a Reed-Solomon code.

The representation of the FNT comes from polynomials and we represents a vector of size n, we define a(x) as the polynomial as

$$\sum_{i=0}^{n-1} a_i x^i \quad \ldots\ldots\ldots\ldots\ldots (1)$$

Thus FNT can be viewed as the evaluation of a(x) on the points which form a geometric sequence.

$$1, r, r^2, \ldots\ldots, r^{n-1}, A_j = a(r^j) \; 0 \le j \le n-1,$$

3.2 Token Computation

Algorithm 1: Token Pre-Computation
1: Procedure
2: Choose parameters l, n and functions f and π;
3: Choose the number of t verification tokens;
4: Choose the number of r indices per verification;
5: Generate random challenge key k_{PRF} and master permutation key k_{PRP}
6: for vector G(j), j←1,n do
7: for round i←1, t do
8: Derive $x_i = f_{k_{SRF}}(i)$ and $y_i = f_{k_{SRP}}(i)$ from kSRP
9: Compute $V_i^{(j)} = \sum_{q=1}^{r} (x_i^q . z^{(j)})$ where
10: end for
11: end for
12: Store all the Vi's locally.
13: end procedure

3.3 Data Integrity Checking

Algorithm 2: Challenge-Response Protocol
1: procedure CHALLENGE (i)
2: user regenerate $x_i = f_{k_{PRF}}(i)$ and $y_i = f_{k_{PRP}}(i)$ from kPRP;
3: user sends {xi, yi} to all the cloud servers;
4: server computeshhhghfdhhhhhhhhhhhhhhhhhh

$$\{R_i^{(j)} = \sum_{q=1}^{r} (x_i^q . z^{(j)})$$
$$where \; z^{(j)} = G^{(j)}[\pi_{y_i}(q)] \; | 1 \le j \le k\}$$

5: server return to $R_i^{(j)}$ to users;
6: if $(R_1^{(1)}, \cdots\cdots R_i^{(m)}) \cdot A = (R_i^{(m+1)}, \cdots\cdots R_i^{(k)})$
7: Accept and ready for next challenge
8: else
9: for j←1, k do
10: if $(R_i^{(j)} != V_i^{(j)})$ then
11 return server j is malfunctioning
12: end if
13: end for
14: end if
15: end procedure

4. Security Analysis

In this we present a formal security analysis for our proposed scheme depending on the detection probability of data corruptions.

Eventually, integrity of all transferred data will be confirmed with high probability. This optional threshold may not be appropriate in certain situations such as highly critical information, but may provide a performance boost from any typical applications relying on outsourced data storage.

Assume that client sets a threshold 't' between 0 and 1 . Whether the individual query is authenticated or not should be decided randomly. It's important to randomize the authentication of queries, or a malicious storage server could predict which queries would be authenticated and provide in correct responses only on unauthenticated queries. The probability of the client detecting that there is atleast one piece of invalid data which can be calculated as $P = 1 - (1 - c * t)$ n, where n is the number of queries that have been performed.

5. Performance Analysis

We now analyze the performance of our auditing scheme. We focus on the encoding cost of file in file preparation as well as the token compute. Our experiment is conducted on a system with an Intel Core 2 processor running at 1.86 GHz, 4GB RAM, 7,200 RPM and Western Digital 520 GB Serial ATA drive. Algorithms are implemented using open-source erasure coding library Jerasure.

5.1 File Encoding

As discussed earlier, in file encoding includes the generation of parity vectors. We consider two sets of different parameters for the FNT based (m,n) Reed-Solomon encoding, both of which work over GF(2^{16}). Fig. 2 shows the total cost for encoding of different file sizes in GB file before outsourcing using Vandermonde Reed-Solomon code and FNT based Reed-Solomon Code. From the figure we can say that FNT based reed-solomon code is faster than Vandermonde Reed-Solomon

Compared to the existing work, it can be shown from Fig. 4 that the file enocding of our scheme is more efficient. This is because in vandermonde Reed-Solomon code[14] performs then multiplications based on Galios Field(GF) whereas our scheme performs FNT based Reed Solomon Eraser code.

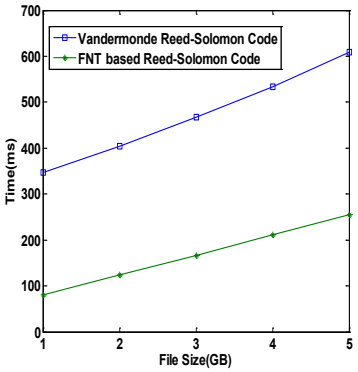

Fig. 2 Total cost for encoding the different file sizes in Vandermonde Reed-Solomon erasure code and FNT based Reed-Solomon erasure code.

5.2 Token Pre-Comutation

In this section, we measure the time for computing tokens in token computation phase. Here, we are using Universal Hash Function (UHF) over the Galios Field(GF). Table 1 shows that computation cost of tokens.

File Size	Proposed Scheme	Existing Scheme
1GB	203.23s	252.21s
2GB	246.63s	294.13s
3GB	283.80s	332.28s
4GB	325.76s	374.43s
5GB	366.32s	415.09s

Table 1. Computation times for different file sizes of above two schemes.

6. Related Work

Juels and Kaliski Jr. [3] describes "proof of retrievability" (POR) model for ensuring the remote data integrity. Their scheme combines spot-checking and error-correcting code to ensure both possession and retrievability of files on service systems. Bowers et al. [11] proposed an better framework for POR protocols that generalizes both Juels and Shacham's work. Later in their consequent work, Bowers et al. [14] extended POR model to distributed systems. Even though, all these schemes are focusing on static data. In their subsequent work, Ateniese et al. [7] described a PDP scheme that uses only symmetric key-based cryptography with lower overhead than previous schemes and allows for block operations. However, their scheme focuses on single server scenario and does not provide data availability guarantee against server failures. The explicit support of data dynamics has been further studied in the two recent work [8] and [9]. Schwarz and Miller [15] proposed to ensure static file integrity across multiple distributed servers.

Very recently, Wang et al. [16] proposed a flexible distributed storage integrity auditing mechanism, utilizing the homomorphic token and distributed erasure-code data. The proposed design allows users to audit the cloud storage with very trivial communication, computation cost and support of dynamic data operations.

7. Conclusion

In this paper we examine the problem of data security in cloud data storage. The integrity and availability of cloud storage service is achieved by flexible distributed scheme that will supports dynamic operations including block update, delete,append and insert. In this paper we rely on FNT based Reed Solomon erasure correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. Our scheme also achieves the integration of storage correctness insurance and data error localization through utilizing the homomorphic token with distributed verification of erasure coded data. Clearly, by observing the performance results our proposed scheme is more efficient against byzantine failures and drastically reduce the computation time and overhead compared to the Reed Solomon erasure coded data.

REFERENCES

[1] C. 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] K. Ren, C. Wang, and Q. Wang, “Security Challenges for the Public Cloud,” IEEE Internet Computing, vol. 16, no. 1, pp. 69-73, 2012. | [3] 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. | [4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable Data Possession at Untrusted Stores,” Proc. 14th ACM Conf. Computer and Comm. Security (CCS’07), pp. 598-609, Oct. 2007. | [5] M.A. Shah, M. Baker, J.C. Mogul, and R. Swaminathan, “Auditing to Keep Online Storage Services Honest,” Proc. 11th USENIX Workshop Hot Topics in Operating Systems (HotOS ’07), pp. 1-6, 2007. | [6] M.A. Shah, R. Swaminathan, and M. Baker, “Privacy-Preserving Audit and Extraction of Digital Contents,” Cryptology ePrint Archive, Report 2008/186, http://eprint.iacr.org, 2008. | [7] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” Proc. Fourth Int’l Conf. Security and Privacy in Comm. Netowrks (Secure Comm ’08), pp. 1-10, 2008. | [8] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing,” Proc. 14th European Conf. Research in Computer Security (ESORICS ’09), pp. 355-370, 2009. | [9] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, “Dynamic Provable Data Possession,” Proc. 16th ACM Conf. Computer and Comm. Security (CCS ’09), pp. 213-222, 2009. | [10] H. Shacham and B. Waters, “Compact Proofs of Retrievability,” Proc. 14th Int’l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (Asiacrypt ’08), pp. 90-107, 2008. | [11] K.D. Bowers, A. Juels, and A. Oprea, “Proofs of Retrievability: Theory and Implementation,” Proc. ACM Workshop Cloud Computing Security (CCSW ’09), pp. 43-54, 2009. | [12] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, “Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing,” IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, 2011. | [13] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, and W. Lou, “Privacy-Preserving Public Auditing for Secure Cloud Storage,” IEEE Trans. Computers, preprint, 2012, doi:10.1109/TC.2011.245. | [14] K.D. Bowers, A. Juels, and A. Oprea, “HAIL: A High-Availability and Integrity Layer for Cloud Storage,” Proc. ACM Conf. Computer and Comm. Security (CCS ’09), pp. 187-198, 2009. | [15] T. Schwarz and E.L. Miller, “Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage,” Proc. IEEE Int’l Conf. Distributed Computing Systems (ICDCS ’06), pp. 12-12, 2006. | [16] Cong Wang, Qian Wang, Kui Ren, Ning Cao and Wenjing Lou, “Toward Secure and Dependable Storage Services in Cloud Computing,” IEEE Tranctions on services computing, vol 5, No. 2, April-June 2012 | [17] J. Hendricks, G. Ganger, and M. Reiter, “Verifying Distributed Erasure-Coded Data,” Proc. 26th ACM Symp. Principles of Distributed Computing, pp. 139-146, 2007.