INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS…..

Local Reconstruction Codes in Cloud

Salitha K. K.[1], **Shibu K. R.**[2]

M.Tech Student (CSE), Mahatma Gandhi University
Viswajyothi College of Engineering and Technology, Muvattupuzha, Kerala, India.
salithakesavan@gmail.com

Assistant Professor (CSE), Mahatma Gandhi University
Viswajyothi College of Engineering and Technology, Muvattupuzha, Kerala, India.
krshibu@rediffmail.com

Abstract – *Cloud computing intendedfor providing software and hardware facilities are provided to users as a service over the internet through a web browser. Economic gains are the foremost driver for the Cloud. User need not worry about the internal details of the storage and physical infrastructure. But, privacy and security risks are high in this case. Hence correctness of data is a prime concern. This work proposesa novel approach for data correctness in cloud storage systems which is a reliable storage system based on Local Reconstruction Codes (LRC). While reconstructing data chunks LRC reduces the number of erasure coding chunks that need to be read, but keeping the storage overhead low. The vital benefits of LRC are that it moderates the bandwidth and input output operations required for repair reads, while still agreeing a major reduction in storage overhead. With the support of Arbitrator this scheme guarantees data error correction along with integrity checking.*

Keywords – *Cloud Computing, Data Integrity, Security, Distributed Storage, Local Reconstruction Code*

1. INTRODUCTION

According to National Institute of Standards and Technology (NIST) Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1].Cloud computing users do not own the physical infrastructure; rather they rent service from a cloud provider. They consume resources and pay only for what they use. Cloud storage systems are housed in facilities called datacenters. Client can communicate with one of the servers. But there can be several datacenters on which data can be stored. Since users don't have to care about the difficultiesof direct hardware management, moving files into the cloud offers great convenience to them.A malicious cloud provider or any other outsider can modify the client's data and hence, the integrity of the data will be lost. Maintaining the correctness of user files is not so easy because the data files tend to be updated frequently by the user.

The coolest approach to provide integrity for data is to replicate it. But evidently, replication uses a lot of storage. The storage cost, or overhead will be high. Erasure coding schemes are the other key solution to address this problem. In erasure coding, a file F consisting of k blocks can be encoded using an(n; k) erasure code to create n codedblocks out of the original k file blocks, and keeps them at n servers on the basis of one encoded block per server. Thus, the original file can be regenerated from any k out of the n servers. When the client discovers modification of one of the encoded blocks, it can make use of the remaining healthy blocks to reconstruct the corrupted block. The desirable properties of an erasure coding methods used in a cloud storage system should possess the following characteristics:

1. Reconstruction of data from minimum number of chunks
2. Less storage overhead

Local Reconstruction Codes (LRC) provides the above properties. In addition, we can analyze the Reed Solomon encoding which is a widely accepted erasure code along with Local Reconstruction Codes.

1.1 RELATED WORKS

Cloud computing make it possible to store large amounts of data. But data integrity is the main problem with cloud storage. Juels et al. [2] described a proper "proof of retrievability" (POR) approach for guaranteeing the remote data integrity. A random linear function based homomorphic authenticator defined by Shacham et al. [3] which does not require much communication overhead. These previous efforts improved by Bowers et al. [4]. Later in their succeeding work, Bowers et al. [7] stretched POR model to distributedsystems as a High Availability and Integrity Layer (HAIL). Still, all these systems are concentrating on staticdata. Any alterations to the contents of the file, evenfew bits, must disseminate through the error correcting code, consequently introducing substantial computation and communication complexity.

A "provable data possession"(PDP) was proposed by Ateniese et al. [5] whichassures possession of file on untrustedstorages based on public key cryptography, therefore ensuring public verifiability. But, when compared with others this approach is suffering from sufficient computationoverhead and can be expensive. Improved PDP scheme is defined by Ateniese et al. [6] based on symmetric key encryption for which overhead is lesser than preceding scheme and permits future modifications. But, this approach concentrates only on single server. Curtmola et al. [9] suggested a solution to this problem by allowing data possession of multiple replicas. Shah etal. [10] kept online storage honestby first scrambling the data then transferring a number of pre-computed symmetrickeyed hashesover the encrypted data tothe arbitrator. But, their scheme only works for encryptedfiles.

2. LOCAL RECONSTRUCTION CODES

If encoding and decoding takes a lot of computational resources then it annoys other operations like encryption, compression and so on.Because of the generality of the Reed-Solomon codes they are widely used in distributed storage systems. Reed-Solomon coding is designed for deep space communication, because mistakes occur frequently in this type of communication.However, the data centers act in a different way from that of deep space communication. Awell-designed and supervised server has very less failure rate which in turn entails that furthermost data files in the data centers are strong, with no failures. Even though some errors are present they will be very less in amount and will last for a short duration. Also error will be repairedas fast as possible and will be brought them to healthy state.

The new coding approach enables data to be reconstructed more rapidly than with Reed-Solomon codes. This is because lesser data chunks must be accessed to regenerate the original data. Only half the chunks are necessary. Moreover, these codes are mathematically simpler than former procedures. The local in the coding technique's name denotes, in the event of a failure, the code required to recreate data is not spread across all the servers.

Data durability is outstanding with these type of codes such that a data fragmentcan be rebuilt with complete accuracy. The reduction in communication overhead helps to decrease the storage cost and it has faster reconstruction costs than previously available codes.

Erasure-correcting code may be used to survive numerous failures in distributed storage systems. Message to be communicated is divided into several blocks. An (m,k) Reed-Solomon erasure-correcting code is used to produce k redundancy parity vectors from m data vectors in such a way that the original m data vectors can be recreated from any m out of the m + k data and parity vectors. Place each of the m + k vectors on different servers, the original data file can tolerate the failure of any k of the m + k servers without any damage, with a storage overhead of k=m [8].

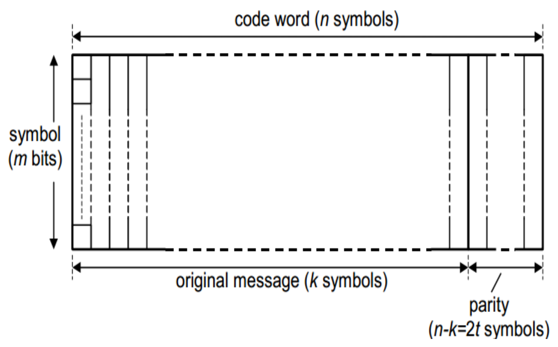

Figure 1: Reed Solomon

As an example [13] consider a (6, 3) Reed-Solomon code with 6 data chunks and 3 parity chunks, such that each parity is calculated from all the6 data chunks. During failure all the 6 chunks are required for the recreating data. Consider reconstruction cost is same as the number of chunks required to rebuild an inaccessible data fragment. Therefore here it is equals to 6.The aim of LRC is to decrease the reconstruction cost.It attains this by calculating some of the parities froma subset of the data chunks.

With the same example LRC creates 4 parities. Global parities p0and p1are computed from all the data chunks. LRC splits thedata chunks into two equal size clusters and computes a local parity for each cluster.

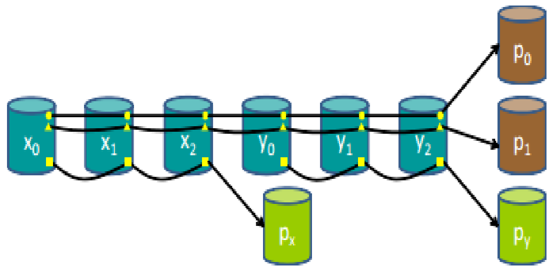

Figure 2: Local Reconstruction

When we analyze this type of codes we can realize that the reconstruction of any single data fragmentneeds only 3 chunks, half the number needed bythe Reed-Solomon code. LRC increases parity by one than Reed-Solomon.
A (k,l,r) LRC divides k data chunks into l groups,with k/l data chunks in each group. It computes onelocal paritywithin each group. Also, it computes r global parities from all the data chunks. Total number chunks (data + parity) is denoted by n where n =k + l + r. Therefore, the normalized storage overhead isn/k = 1 + (l + r)/k. So a (6, 2, 2) LRC has storage cost of 1 + 4/6 = 1.67x, as depicted in Figure 3 [13].

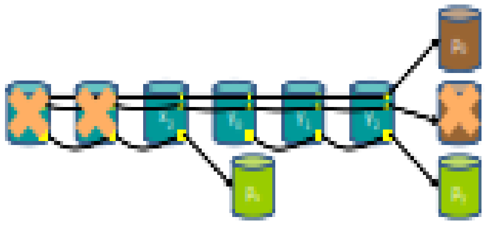

Figure 3: Handling Failures

We must select the coding equations suchthat LRC can attain the Maximally Recoverable (MR)property [14], which means it can interpret any failure which is information-theoretically decodable. Given the (6, 2, 2) LRC, with 4 parity chunks and can survive up to 4 failures. But, it is not Maximum Distance Separable and thus cannot tolerate arbitrary 4 failures. For example, consider x1, x2, x3and px are failed. This is non-decodable because x cannot be reconstructed from any of the global or local parities. Failure patterns that is likely to recreate aresocalled information-theoretically decodable.
In common, the vital properties of a (k,l,r) LRC are: i) single data chunk damage can be interpreted from k/l chunks; ii) arbitrary failures up to r + 1 can be decoded. Based on the subsequent theorem, these properties enforce a lower bound on the number of parities [12].

Theorem 1: For any (n,k) linear code (with k data symbols and n − k parity symbols) to have the property:

1. Arbitrary r + 1 symbol failures can be decoded;
2. Single data symbol failure can be recovered from [k/l] symbols.

The following condition is necessary:
$n - k \geq l + r$

Since the number of parities of LRC meets the lowerbound exactly, LRC accomplishes its properties with the minimal number of parities.

Before uploading the file into the provider's site owner of the file performsLRCencoding and outcome of this procedure will be a collection of data chunks with local and global parities. After this stage cloud user calculates verification tokens on separate chunks. Blind the parity information before transferring it to arbitrator's site. Cloud user then scatters all encoded chunksto the datacenters. In future user can challenge the cloud server and data correctness is verified by the auditor. Intervention of auditor lessens the computation overhead of the user. This approach also promotesfuture modifications to the file in an efficient and well-organized manner [11].

3. CONCLUSIONS

Erasure coding is important to lessen the expense of cloudstorage. Most important factor in these codes is fast reconstruction of offline data chunks. A (k,l,r) LRC divides k data chunks into l local groups and encodes l local parities, one for each local group, andr global parities. Single data chunk failure can bedecoded from k/l chunks within its local cluster.Additionally, LRC reaches Maximally Recoverableproperty. It accepts up to r + 1 arbitrary fragment failures. It also tolerates failures more than r +1 (up to l +r), as long as those are information-theoretically decodable.As a final point, LRC provides low storage overhead. For the reconstruction, LRCneeds the minimum number of parities.Local Reconstruction Codes as a wayto decrease the number of chunks that need to be readfrom to perform this reconstruction and hassimilar latency for small I/Os and improved latency for large I/Os.

REFERENCES

[1] **T. GranceP. Mell, , "The NIST definition of cloud computing National Institute of Standards and Technology (NIST) 2009**

[2] **Juels andJ. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files,"** ***Proc. of CCS '07*****, pp. 584–597, 2007.**

[3] **H. Shacham and B. Waters, "Compact Proofs of Retrievability,"** ***Proc. of Asiacrypt '08*****, Dec. 2008.**

[4] **K. D. Bowers, A. Oprea, A. Juels, and "Proofs of Retrievability: Theory and Implementation,"** ***Cryptology ePrint Archive*****, Report 2008/175, 2008, http://eprint.iacr.org/.**

[5] **G. Ateniese, R. Curtmola, R. Burns, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores, "** ***Proc. Of CCS '07*****, pp. 598–609, 2007.**

[6] **G. Ateniese, R. L. V. Mancini, D. Pietro, and G. Tsudik, "Scalable and Efficient Provable Data Possession,"** ***Proc. of SecureComm '08*****, pp. 1–10, 2008.**

[7] **K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for**

Cloud Storage," *Cryptology ePrint Archive*, Report 2008/489, 2008, http://eprint.iacr.org/.

[8] J. S. Plank and Y. Ding, "Note: Correction to the 1997 Tutorial on Reed-Solomon Coding," *University of Tennessee, Tech. Rep.* CS-03-504, 2003.

[9] R. Curtmola, R. Burns, O. Khan, and G. Ateniese, "MR-PDP: Multiple-Replica Provable Data Possession," *Proc. of ICDCS '08*, pp. 411–420, 2008.

[10] M. Baker, M. A. Shah, J. C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," *Proc. 11th USENIX Workshop on Hot Topics in Operating Systems (HOTOS '07)*, pp. 1–6, 2007.

[11] Q. Wang, 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.

[12] C. Huang, P. Gopalan, H. Simitci, and S. Yekhanin, "On the Locality of Codeword Symbols," *Allerton*, 2011.

[13] Cheng Huang, Huseyin Simitci, YikangXu, Aaron Ogus, Brad Calder, Parikshit Gopalan, Jin Li, and Sergey Yekhanin, "Erasure Coding in Windows Azure Storage," *USENIX ATC* 2012

[14] C. Huang, M. Chen, and J. Li, "Pyramid Codes: Flexible Schemes to Trade Space for Access Efficiency in Reliable Data Storage Systems," *Proc. of IEEE NCA, Cambridge, MA*, Jul. 2007.