Reliable and Secure Data Availability in Distributed Cloud Data Storage using Optimal Exact – Regenerating Codes

Malay Kumar
Research Scholar
National Institute of Technology
Raipur, CG-492010

Aprna Tripathi
Associate Professor
Mangalayatan University
Aligarh, India

Manu Vardhan
Assistant Professor
National Institute of Technology
Raipur, CG-492010

ABSTRACT

One of many advantages of cloud computing is to provide reliable data storage facility. The cloud computing provides seamless access to storage facility to the client to upload, download and modifies unlimited amount data. However, at the same time, outsourcing data to the third party cloud storage system is a great cause of concern to the client. The client loses physical control of the data which compromises the security, reliability, and confidentiality of data. This paper proposes a new framework for reliable and secure data storage which ensure data security, reliability and availability using Optimal Exact – Regenerating Codes [1]. This framework is different from existing approach of data redundancy for ensuring data availability and reliability. To, ensure the data availability our method relies on multiple cloud service providers (CSP). Each CSP is view as individual disks of RAID where some parts of client data are stored. Since none of the CSP have complete access to client data so, the individual CSP cannot breach the client data. The proposed framework is compared with traditional erasure coding such as Reed- Solomon Codes (RSC). RSC causes much higher repair cost for failed disk and even higher access latency.

Keywords

Regenerating Codes, Erasure Codes, Data Availability, Data Reliability, Data Security, Cloud Storage, Cloud Computing.

1. INTRODUCTION

Cloud computing the longed dreamed vision of computing as a utility, enables great ease to business and individuals who unwilling or unable to procure IT [2]. This promising computing provision most of the services online what we releasing on the desktop computer till now. The services such as email, word processing, spreadsheets, presentations, audio, photos, videos offer as SAAS, development platforms such as Java, Python; GO offered as PAAS, and the architecture to accommodate such services IAAS [3].

Today, every computing node generates tremendous amount personal and business related data. Therefore, the traditional way of data storage unable to cope up with tremendous storage need. They have to rely on some third party vendor. Cloud is capable of fulfilling their need and provides an efficient and cost-effective data storage solution. Cloud computing provides seamless access to their data center to support the need for business and individuals [4]. However, this promising computing paradigm inertly brings security, privacy, and availability concerns that make a client reluctant to use cloud services.

The private sensitive that may be personal identifiable health records, business records, financial records, various market trends, etc. When this dataset goes beyond the possession of client, data reliability became a big question. The end-user which really on the services need to access the data whenever they want them which push the cloud to deliver the storage services with greater reliability without or minimum downtime. CSP guarantees to provide highly reliable service in their SLA, but there are many occasions of a cloud outage. A study reveals that an average 7.5HRS of downtime of cloud outage per CSP over a year, which is 99.9% availability. However, the many of the business critical application requires 99.999% availability of access facility i.e. 1HRS of downtime in an entire year, which further pushes the expectation for highly reliable cloud service. Cloud stores a substantial amount of data in commodity disk inside their data centers. The commodity disks are very likely to failure even if this disk did not get to failure the hosting node may get a failure as a consequence the data became inaccessible. Thus, replication is not capable of avoiding cloud outage alone. Thus, CSP must consider the question of data reliability very carefully because a minimal cloud outage collaterally damages many businesses dependent upon a cloud. This expectation pushes the CSP to deliver highly reliable cloud services without any downtime and handle the failure which affecting minimum users [5].

In this paper, we investigate the Optimal Exact – Regenerating Codes to address the reliability and availability issue in cloud storage. Our aim is to achieve low access latency time, low CPU requirement for encoding and decoding, high input-output throughput with minimal overhead, i.e., minimum storage and bandwidth utilization.

2. LITERATURE REVIEW

Data replication is a natural choice to increase the reliability of storage systems since data replication tolerates the data loss as long as there exists one replica available [6][7]. However, this system suffers from highly inefficient space efficiency and when we talk about cloud computing where the clients store a massive amount of data on cloud server which makes the overall system inefficient. One optimal solution is erasure coding; the solution gives the reliability of redundancy with the additional advantage of space efficiency. The one main disadvantage of this method is that the process consumes CPU time in encoding and decoding operation which makes this approach ill-suited for the resource-constrained client. However, the method provides the flexibility that it gracefully tolerates the certain number disk failures. However, since there is a broad range of cloud user this method is not useful for all users besides even resourceful client do not want to waste their CPU resources for decoding and encoding operation. The Reed-Solomon codes were first introduced as error correcting code. Now, they are being used as a solution for many online storage systems [8]. RS codes encode the data segment and upload to individual cloud service providers. Each node stored the fragment of data as a single symbol over the finite field . The drawback of this method is that if the individual nodes are restricted to perform only linear operation over , the total amount of data download needed operations over to repair a failed node can be no smaller than *B*, the size of the entire file. However, two main drawbacks conventionally prevent erasure coding from being practical and popular in cloud storage. First, to read or write data, the system needs to encode or decode data, leading to a high access latency and a low access throughput due to the CPU limit. Second, though erasure coding stores data as multiple coded blocks, when one coded block gets lost, the system must access multiple coded blocks that are sufficient to recover the data [9]. Another solution is Reed-Solomon Codes (RS) [10]. This technique is more efficient than the replication and able to tolerate a certain number of disk failures without losing the space efficiency [11]. Network coding is a more advanced version of traditional network routing. In convention routing the intermediate node stores and forward the information. Unlike this the network coding allows the intermediate nodes to perform computation on the data received from the previous nodes. Network coding is increasing the robustness of network dynamics. Thus, reduce the bandwidth, power, efficiency requirement for node repair. A significant amount of research work focuses on providing reliable data storage system in distributed environment [11]. The work in [12], [13] provide a quantitative comparison of redundancy-based approach to erasure coding based approach. Erasure coding such as Reed-Solomon codes can provide even more flexibility such that any number of disk failures under a certain threshold can be tolerated. An online secure storage service, uses Reed-Solomon codes to ensure data integrity, by encoding data with Reed- Solomon (RS) codes after encryption [8]. RS codes encode the data segment and upload to individual cloud service providers. Each node stored the fragment of data as a single symbol over the finite field. The drawback of this method is that if the individual nodes are restricted to perform only linear operation over , the total amount of data download needed operations over to repair a failed node can be no smaller than *B*, the size of the entire file. However, two main drawbacks conventionally prevent erasure coding from being practical and popular in cloud storage. First, to read or write data, the system needs to encode or decode data, leading to a high access latency and a low access throughput due to the CPU limit [14]. Second, though erasure coding stores data as multiple coded blocks, when one coded block gets lost, the system must access multiple coded blocks that are sufficient to recover all the data [9]. Recently, HI&bhelaro [15] presented a paper which uses erasable coding and address reliability, security, and performance of storage system. This approach was apart from the traditional way to achieve data reliability at server side through redundancy rather it is a software based approach at end-user side. The end-user divides the data into small data pieces then encode them, the number of CRC pieces decides by an optimal problem then finally upload the data and CRC at multiple CSP. At any point of time the end-user able to recover its data. Dimakis et al., [11] presents another approach for data reliability in its pioneer paper for data reliability. They first time given the concept for regenerating codes. Regenerating codes very efficient they the combination of best this in erasure codes and replication method that is storage efficiency of erasure codes and communication efficiency of replication. These codes are efficiently repairs the failed node as compare to erasure codes. Rashmi et al. [1] presents an optimal exact regenerating codes. A regenerating code , where the data can be recovered connecting any of the nodes, and repair of the failed node require to connect d nodes, The previous constriction confined the case for . Now they present MBR construction for any value of , and MSR construction for all , using a product matrix framework. The focus of this paper is to provide an effective framework using the regenerating codes. Unlike erasure coding which suffers from large latency time for encoding and decoding. The regenerating codes are very efficient; the access latency time reduces to replication method, and the storage efficiency reduces to erasure coding [16].

3. CHALLENGES

There is three main challenge in the development of storage system that is security, reliability, and availability of the stored data. The system design for cloud computing comes with own set of challenges. The storage system must be capable of manage the vast amount of data with a large amount of I/O operations. Achieving the security, reliability, and availability is a challenging task for CSPs since data stored on a large number of commodity disk into data centers is prone to failure. This challenges attracted a significant number researcher to provide an efficient and novel technique for massive data storage on the cloud. The storage system achieves the data reliability by replication in which data the server stores the data at multiple locations the scheme suffers low storage efficiency and not practical in cloud computing. Erasure coding is providing an option for secure and reliable storage system in which the client divides the data into multiple data pieces and the total pieces of data being constructed and uploaded to multiple cloud service providers. Any k of them are enough to reconstruct the data file. However, this scheme is suffers from high access latency, a low access throughput due to the CPU limitation and high bandwidth requirement for failure repair. Regenerating codes are achieving the CPU limitations and save bandwidth by not transferring data that are unnecessary to the particular newcomer.

4. REGENERATING CODES

Regenerating codes similar to Galois field work over finite field . The codes are specified as vector alphabets over α symbols. Linear operations over in this case, permit the transfer of a fraction of data stored on particular node. A part from this new parameter α, two other parameters *d* and *β* are associated with regenerating codes. Under the definition of regenerating codes introduced in [11], a failed node is permitted to connect to an arbitrary set of *d* of the remaining nodes while downloading symbols from each node. This process is termed as regeneration and, the total amount *dβ* of data downloaded for repair purposes as the *repair bandwidth*. Further, the set of *d* nodes aiding in the repair are termed as helper nodes. Typically, with a regenerating code, the average repair bandwidth is small compared to the size of the file *B*. It will be assumed throughout the paper, that whenever mentioned is made of regenerating code, the code is such that are the minimum values under which data-reconstruction and regeneration can always be guaranteed. This restricts the range of *d* to

$$k \leq d \leq n-1 \qquad (1)$$

is the total number of nodes, client can regenrate the data connecting any number of nodes, for the purpose to solve the problem of failure node, a node could be repaied using d number of nodes where . The regenrating code over a finite field represented using a number of parameters , where regard as primary parameter while regard as secondary paramer [1].

Table 1. Notations

B	Total file size
	Finite field of order q
n	Total number of nodes/CSPs in the network
d	Total number of remaining nodes/CSPs
k	Arbitrary set of remaining nodes/CSPs where
	Encoding Matrix
α	Coding Symbols over the finite field
β	Coding Symbols over the finite field

The parameters of regenerating codes must satisfy the following relation.

$$B \leq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\} \quad (2)$$

It is desirable to minimize both α as well as β since, minimizing α results in a minimum storage solution, while minimizing (for fixed) β results in a storage solution that minimizes repair bandwidth. The parameter are complementary to each other. Therefore, the client need to minimize both the parameter values which is a tradeoff so it require make a balance between the value of . The two extrema are defined as MSR and MBR.

Minimum Storage Regeneration

The two exterma are defined as

$$\alpha = B/k$$

$$\beta = {}^{B}/_{k(d-k+1)} \quad (3)$$

α and β are complimentary to each other and dependent in the case of MSR first diminishing the value of α and then β

Minimum Bandwidth Regeneration (MBR)

When reversing the parameter, which is minimizing the β first and then minimizing α.

$$\beta = {}^{2B}/_{k(2d-k+1)}$$

$$\alpha = {}^{2dB}/_{k(2d-k+1)} = d\beta \quad (4)$$

Striping of Data

β=1 in the terminology of distributed system called stripping of data. The equations for MSR and MBR respectively, when $\beta=1$.

For MSR Codes

$$\alpha = d - k + 1$$

$$B = k(d-k+1) \quad (5)$$

For MBR Codes

$$\alpha = d$$

$$B = kd - \binom{k}{2} \quad (6)$$

The for more information about MBR and MSR see in [1]

5. SECURE AND RELIABLE DISTRIBUTED STORAGE SYSTEM

This paper uses a product matrix framework for reconstruction and regeneration of data. Under this context, each code-word in the distributed storage code can be represented by an code matrix C whose i^{th} row contains the contains the α symbols stored by the i^{th} node. Each code matrix is the product of encoding matrix and an message matrix M where Encoding matrix chosen as vandermonde matrix

$$C = \psi M \quad (7)$$

5.1 Product-Matrix MBR Code Construction

Let be a $(k * k)$ matrix constructed so that the $\binom{k+1}{2}$ entries in the upper-triangular half of the matrix are filled up by $\binom{k+1}{2}$ distinct message symbols drawn from the set $\{u_i\}_{i=1}^{B}$. The $\binom{k}{2}$entries in the strictly lower-triangular portion of the matrix are then chosen so as to make the matrix S a symmetric matrix. The remaining $k(d-k)$message symbols are used to fill up a second $(k*(d-k))$matrix T. The message matrix M is then defined as the $(d*d)$ symmetric matrix given by

$$M = \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix} \quad (8)$$

The symmetry of the matrix will be found to be instrumental when enabling node repair. Next, define the encoding matrix ψ to be any $(n*d)$ matrix of the form

$$\psi = [\phi \quad \Delta] \quad (9)$$

Where $\phi\ and\ \Delta$ are $(n*k)$ and $(n*(d-k))$ matrices respectively, chosen in such a way that; any d rows of ψ are linearly independent; any k rows of ϕ are linearly independent[13].

5.2 Product-Matrix MSR Code Construction

At the MSR point with $d = 2k-2$ we have

$$\alpha = d - k + 1 \quad (10)$$

When applying the value of d,

D==2 α.

Also

$$B = k(d-k+1)$$

We define the *(d* α)* message matrix M as

$$M = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \quad (11)$$

Where$S_1$and S_2 are $(\alpha * \alpha)$ symmetric matrices constructed such that the $\binom{\alpha+1}{2}$ entries in the upper-triangular part of each of the two matrices are filled up by distinct message symbols. Thus, all the message symbols are contained in the two matrices $S_1$and S_2. The entries in the strictly lower-triangular portion of the two matrices and are chosen so as to make the matrices and symmetric.

Next,we define the encoding matrix ψ to be the $(n*d)$ matrix given by

$\psi = [\phi \quad \wedge \phi]$
(12)

where is ϕ an $(n * \alpha)$matrix and is $\wedge$ an $(n * n)$ diagonal matrix. The elements ψ of are chosen such that the following conditions are satisfied[13]:

1 Any $d$rows of ψ are linearly independent;
2 Any α rows of ϕ are linearly independent;
3 The n diagonal element of $\wedge$are distinct.

6. RESULT ANALYSIS AND DISCUSSION

To demonstrate the feasibility as well as the performance of our approach, we developed a prototype secure and reliable distributed cloud data storage application. We adopt three different cloud storage services supported by major CSPs to store our data pieces in the cloud. The selected cloud storage services are Amazon S3, Google App Engine, and Core Dropbox APIs with free user accounts. The application was running on a Windows machine with a 3.40 GHz Intel Core i7 processor and 8.00 GB of RAM. For each experiment in our case study, we repeat it at least ten times to get the stable system performance for each particular setting. To analyze the performance of our approach, we selected multiple file type varying in their size. The comparison matrix with the previous algorithm is presented in Table 2. The comparison matrix submitted for each file type with the current state of art implementation [15].

For the purpose to demonstrate the practical usage of the method data files is first converted to the data matrix. The data matrix and further its corresponding vandermonde matrix. The matrix size of the vandermonde matrix is decided by the size of the data matrix. Then produce the product matrix framework. Each row of the product matrix framework is send to individual CSPs. To, decide which cloud to transfer the row (partial message) using linear programming optimal problem.

Table 2. Latency Analysis for Encoding and Decoding

Input File Format	File Size in MB	Enc Time [15]	Dec Time [15]	Enc Time Sec	Dec Time Sec
Video	156	10.16	7.56	8.32	6.51
	317	22.16	15.27	18.73	12.58
	700	32.14	28.53	29.69	26.31
	800	35.28	31.37	31.90	30.10
	1024	51.61	49.99	50.12	48.83
Pdf	2.4	0.136	0.119	0.113	0.091
	2.6	0.151	0.141	0.118	0.110
	2.9	0.169	0.139	0.120	0.096
	10.5	0.612	0.535	0.324	0.218
	16.7	0.974	0.661	0.713	0.512
Tar	46	2.13	1.92	1.97	1.87
	70.5	3.82	3.45	2.91	2.76
	128	5.31	4.97	5.12	4.78
	217	14.105	13.47	12.23	11.53
	286.6	22.629	20.13	21.32	19.94
Docx	4	0.2833	0.196	0.184	0.153
	4.4	0.3116	0.2075	0.295	0.283
	6.5	0.4604	0.397	0.423	0.382
	8.1	0.5737	0.5456	0.518	0.495
	9.2	0.6516	0.5468	0.583	0.495

For example, let take a video file of 156 MB. First, convert this video file into data file then generates the required dimension of vandermonde matrix produces a product matrix. In the result analysis between the method proposed by Xu & Bhalerao [15] the time required form encoding and decoding our method which comprises regenerating codes takes lesser time than this method. As the size of video file increases that is from 156 MB to 1024 MB the average time for encoding requires 27.75 sec while decoding method takes 24.86 sec. The time requires for Pdf files takes only 0.319 sec for encoding and 0.205 decoding. Similarly, for Tar file 8.78 sec in encoding and 8.176 for decoding operation. For docx file the operation takes 0.37 encoding operation sec and 0.36 sec in decoding operation. The comparison presented here demonstrates that this is the actual latency time in addition with the upload and download time for the data whenever it accesses the cloud service providers. This optimal code regeneration scheme perform better in comparison to previous erasure code Reed-Solomon Codes. The scheme equally provide the security, reliability and availability of data.

7. CONCLUSION

In, this paper we present a study and application of optimal exact regeneration codes. The paper also present a comparison analysis of encoding and decoding time between the RS codes and optimal exact regeneration codes. The results are very promising and indicate that this implementation is very useful in real time scenarios. Besides the above fact, the implementation keeps the large dataset of client secure, reliable and available anytime anywhere on-demand.

8. REFERENCES

[1] K. V. Rashmi, N. B. Shah, and P. V. Kumar, “Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction,” *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5227–5239, 2011.

[2] R. Ranchal, B. Bhargava, L. Ben Othmane, L. Lilien, A. Kim, M. Kang, and M. Linderman, “Protection of identity information in cloud computing without trusted third party,” *Proc. IEEE Symp. Reliab. Distrib. Syst.*, pp. 368–372, 2010.

[3] A. Skendzic and B. Kovacic, “Microsoft Office 365 - cloud in business environment,” *MIPRO, 2012 Proc. 35th Int. Conv.*, pp. 1434–1439, 2012.

[4] S. Marston, S. Bandyopadhyay, and a Ghalsasi, “Cloud Computing - The Business Perspective,” *2011 44th Hawaii Int. Conf. Syst. Sci.*, pp. 1–11, 2011.

[5] S. Weil and A. Leung, “Rados: a scalable, reliable storage service for petabyte-scale storage clusters,” *PDSW ’07 Proc. 2nd Int. Work. Petascale data storage held conjunction with Supercomput. '07*, pp. 35–44, 2007.

[6] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, “Erasure Coding in Windows Azure Storage,” *Atc*, p. 12, 2012.

[7] S. Jiekak, A. M. Kermarrec, N. Le Scouarnec, G. Straub, and A. Van Kempen, “Regenerating codes: A system perspective,” *Proc. IEEE Symp. Reliab. Distrib. Syst.*, pp. 436–441, 2012.

[8] S. Distributed, C. Data, and S. Using, “Int ’ l Journal of Software Engineering and Knowledge Engineering Reliable and Secure Distributed Cloud Data Storage Using Reed-Solomon Codes.”

[9] J. Li and B. Li, “Erasure coding for cloud storage systems: A survey,” *Tsinghua Sci. Technol.*, vol. 18, no. 3, pp. 259–272, 2013.

[10] S. B. Wicker and V. K. Bhargava, “An Introduction to Reed-Solomon Codes,” *Reed-Solomon Codes Their Appl.*, pp. 1–16, 1999.

[11] A. G. Dimakis, P. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” *Inf. Theory, IEEE Trans.*, vol. 56, no. 9, pp. 4539–4551, 2010.

[12] H. Weatherspoon and J. D. Kubiatowicz, “Erasure coding vs. replication: A quantitative comparison,” in *International Workshop on Peer-to-Peer Systems*, 2002, pp. 328–337.

[13] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. M. Voelker, “Total Recall: System Support for Automated Availability Management.”

[14] J. S. Plank, “Erasure codes for storage systems: A brief primer,” *Usenix Mag.*, vol. 38, pp. 44–51, 2013.

[15] H. Xu and D. Bhalerao, “A Reliable and Secure Cloud Storage Schema Using Multiple Service Providers,” pp. 116–121.

[16] A. G. Dimakis and K. Ramchandran, “A Survey on Network Codes for Distributed Storage,” vol. 99, no. 3, pp. 1–13, 2011

.