International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878, Impact Factor (2014): 3.05

Proxy Based Backup in a Multiple Cloud Storage with Depreciated Repair Traffic

Abhilatha P[1], Manasa K[2], Bhagya M[3]

[1]Student, Nagarjuna College of Engineering, Department of Computer Science Bangalore, India
[2]Student, Nagarjuna College of Engineering, Department of Computer Science Bangalore, India
[3]Asst.Professor, Senior Grade, Nagarjuna College of Engineering, Department of Computer Science, Bangalore, India

Abstract: ***In this paper demonstrates the use of proxy-based distributed storage in cloud. We are providing implementable design of how to regenerate the lost data when cloud storage failed permanently. Our Scheme achieves cost-efficient repair by using Functional Minimum Storage Regenerating (FMSR) codes which is network coding based storage scheme with reduced repair traffic and incur less monetary cost.***

Keywords: Fault tolerance, FMSR, erasure codes, repair traffic

1. Introduction

Cloud computing is gaining extensive popularity because of its intrinsic resource-sharing, flexible storage capacity, and pay as you go characteristics. Its usage is stretching worldwide in various fields like Military, Education, Research, Business, public organizations and many other [1]. It has been emerged as a intelligent solution in order to provide cheap and easy way to externalized IT resources.

Cloud users can enjoy high-quality services provided by various cloud service providers (CSP) such as Amazon, Redhat, Google with the help of their powerful data centers and also can save significant investments on their local infrastructures. CSP facilitates various scalable IT services [2]. Data storage is the most fundamental services offered by cloud providers along with the on-demand remote backup solution.

In general there are two types of cloud storage failure: transient failure and permanent failure.

Transient failure is the short term failure of cloud and will return to its normal condition after some time. There will be no data loss.

Permanent failure is long term failure means outsourced data will become unavailable permanently. There are several situations like data-center outages in disaster, malicious attacks where permanent cloud failure occurs. There are many real-life cases [3][4] where data in cloud had lost accidently. However usage of single-cloud storage leads to single point of failure.

The plausible solution is to stripe data across multiple storage in a cloud [5]. It is advantageous in terms of security too. By exploiting the diversity of multiple storage, we can improve the fault tolerance of a cloud. When a cloud failed, it is essential to activate repair to maintain fault-tolerance and data redundancy [6]. Regeneration of lost data is achieved by retrieving data from other existing survived storage nodes over the network and reconstructs the lost data. Moving an enormous amount of data across will introduce significant monetary costs. Hence it is essential to minimize the repair traffic as much as possible (i.e., during repair the amount of data that is being transferred over the network).

In this paper, we present the proxy-based storage system design and implementation of a designed for providing fault-tolerant storage system in cloud [7]. Our main focus is on minimizing repair traffic which is achieved by FMSR code implementation [8]. It consumes same storage capacity as other traditional erasure coding schemes but uses less repair traffic when recovering a single-cloud failure only.

2. Related Work

The simplest form of redundancy is replication (i.e. keeping the duplicate of data) [9]. In this method files are divided into blocks and each block is striped across distributed storage nodes in a cloud and replica of each block is generated. If any node fails then simply copy the replica of that file from healthy node is used to reconstruct original file. But it requires often unnecessarily and unsustainably high expense. The storage cost for replication based system is very high.

Another regenerating code is studied extensively in which is widely used by cloud service providers at present namely Reed-Solomon based Raid-6[10] is shown in Figure 1.

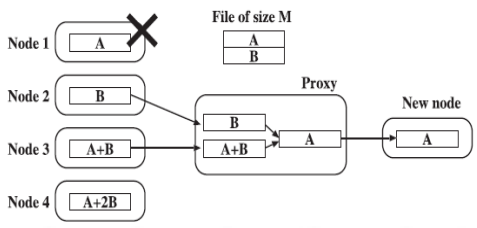

Figure 1: Reed Solomon code

In this scheme file of size M is divided into k fragments stored in a different storage nodes. Size of each fragment is M/k. If any fragment is unavailable, original file can be recovered from remaining fragments. There is no focus on reducing repair traffic. It is of high monetary cost.

3. Proposed Scheme

Our proposed system demonstrates the FMSR code implementation in a proxy based distributed storage system which maintains fault tolerance and also minimizes the repair traffic while recovering from the single cloud failure. It will consume the same storage cost as in traditional erasure coding schemes based on Reed-Solomon.

The FMSR code implementation is as shown in Figure 2

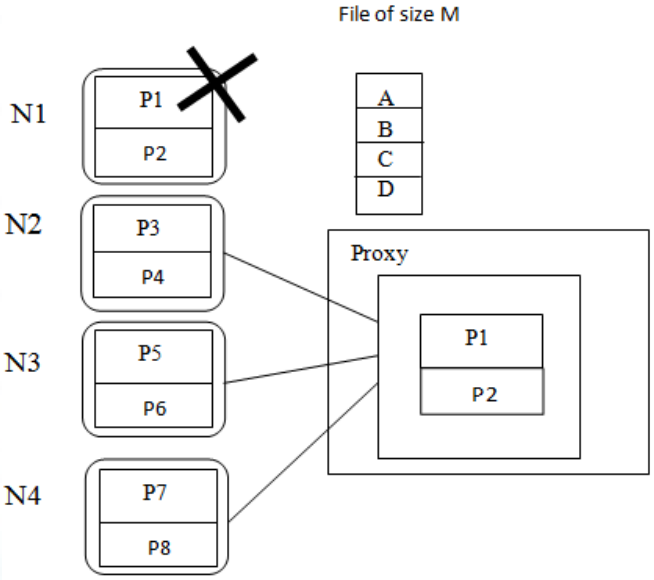

Figure 2: FMSR code

Any file uploaded by cloud user is divided into four native chunks. By using these native chunks eight distinct code chunks P1, P2. . . P8 is constructed by different linear combinations [11] of the native chunks as shown in Figure 3 .

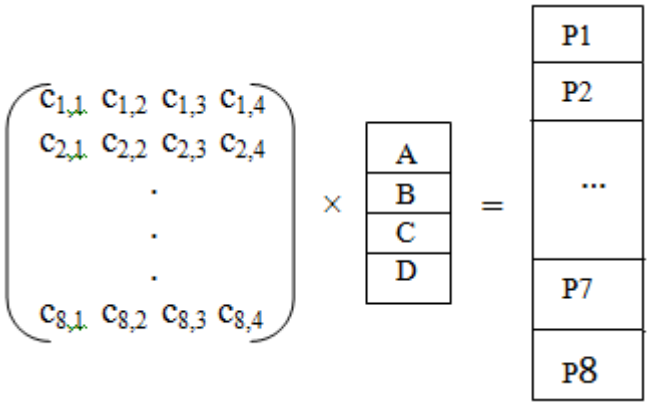

Figure 3: Formation of code chunks

The Encoding matrix which is generated will be multiplied with the each block and code chunks will be generated i.e p1, p2….p8.The size of each code chunk is M/4 which is same as each native chunks. Suppose Node N1 is down then the proxy regenerates lost data by collecting one code chunk ex. P3or p4, p5 or p6, p7 or p8 randomly from each surviving node each of size M/4 i.e. total 3M/4 amount of data is used to regenerate original file. Its repair traffic is 3M/4 whereas in the existing system repair traffic is M. In our proposed system repair traffic is reduced by 25% compared to existing system.

4. Scope

Our implementation of a proxy-based storage system with FMSR code provides fault-tolerance in multiple-cloud storage. It also achieves the cost-effective repair in case of permanent single-cloud failure. One key design feature of our FMSR codes is that we relax the encoding requirement of storage nodes, while preserving the benefits of network coding during repair.

5. Implementation

Implementation of Proposed System is divided into four modules: user, file upload, download and repair.

User: This module consists registration, login, group-creation, join request and accept request operations. User first register by giving input username, password and email. Once register he can update his details after login. He should create a group before upload data. Other users can register under this group. Once user confirmed other users can access the data uploaded into this group.

File Upload: User can upload file using this module. When file is uploaded, the file is divided into equal size and stores it into buffer. This buffer is called native chunks.

The file upload operation consists following operations.

1. Each code chunk is encoded and forms a coded chunk. Code chunk is denoted as P.
   Each Pi is formed by a linear combination of the native chunks.
2. Encoding matrix (EM) is created using Galois Fields.
3. Create Meta data object embed with EM.
4. Upload coded chunks and metadata object into n nodes.

File Download:

File downloaded by using following steps.

1. Download the metadata object that contains the encoding coefficient vector (ECV).
2. Select k of n storage nodes and get the code chunks from k nodes.
3. Generate the square matrix using ECVs and multiply square matrix with code chunks to generate native chunks.
4. Merge Native chunks, decode it and send for download.

Repairs: Corrupted node data repair done by following steps.

1. Download metadata object from surviving nodes. Get the EM from metadata object.
2. Select one ECV from selected surviving node. Generate a repair matrix (RM)
3. Generate new ECV by multiplying RM with selected ECV in step 2.
4. Generate the new encoded matrix. And new coded chunks.

5. Upload new encoded matrix and coded chunks to new node.
6. Evaluation

We define the repair traffic as the amount of outbound data being downloaded from the other surviving clouds during the single-cloud failure recovery. We seek to minimize the repair traffic for cost-effective repair. To generalize double-fault-tolerant FMSR codes for n storage nodes, we divide a file of size M into 2(n-2) native chunks, and use them to generate 2n code chunks. Then, each node will store two code chunks of size M/ 2(n-2) each. Thus, the total storage size is Mn/(n-2). To repair a failed node, we download one chunk from each of the other n-1 nodes, so the repair traffic is M(n-1)/2(n-2). In contrast, for RAID-6 codes, the total storage size is also Mn/n-2, while the repair traffic is M. Our proposed system reduces repair traffic by 25% of repair traffic, compared to Reed-Solomon based erasure codes.

6. Conclusion

We present a proxy-based, multiple-cloud storage system that practically addresses the reliability of today's cloud backup storage. FMSR codes, which regenerates new parity chunks during repair subject to the required degree of data redundancy. Our FMSR implementation eliminates the encoding requirement of storage nodes (or cloud) during repair, while ensuring that the new set of stored chunks after each round of repair preserves the required fault tolerance. Our proposed system not only provides fault tolerance in storage, but also allows cost-effective repair when a cloud permanently fails.

References

[1] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," Univ. California, Berkeley, Tech. Rep. UCBEECS-2009-28, Feb. 2009.
[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing, " Comm. the ACM, vol. 53, no. 4, pp. 50-58, 2010.
[3] Amazon Web Services, "AWS Case Study: Backupify," http://aws.amazon. com/solutions/case-studies/backupify/, 2013.
[4] M. Arrington, "Gmail Disaster: Reports of Mass Email Deletions, "Dec.2006.http://www.techcrunch.com/2006/12/28/gmail-disaster-reports-of-assemail deletions
[5] R. Ahlswede, N. Cai, S.-Y.R. Li, and R.W. Yeung, "Network Information Flow, " IEEE Trans. Information Theory, vol. 46, no. 4, pp. 1204-1216, July 2000.
[6] Gupta Sarika, Sangita Rani Satapathy, Mehta Piyush and ripathy Anupam, "A Secure and Searchable Data Storage in Cloud Computing", 3rd IEEE International Advance Computing Conference (IACC), 2013, page 106-109.
[7] Liu Hao, Dezhi Han, "The study and design on secure-cloud storage system", In IEEE society, 2011, page 5126-5129.
[8] A.G. Dimakis, P.B. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems" IEEE Trans. Information Theory, vol. 56, no. 9, pp. 4539-4551, Sept. 2010.
[9] M.O. Rabin, Simple replication based distributed system," ACM, vol. 36, no. 2, pp. 335-348, Apr. 1999.
[10] J.S. Plank, "A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-Like Systems," Software—Practice & Experience, vol. 27, no. 9, pp. 995-1012, Sept. 2005
[11] K.M. Greenan, E.L. Miller, and T.J.E. Schwarz, "Optimizing Galois Field Arithmetic for Diverse Processor Architectures and Applications," Proc. IEEE Int'l Symp. Modeling, Analysis and Simulation of Computers and Telelcomm. Systems (MASCOTS '08), 2008