Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 5.258

IJCSMC, Vol. 5, Issue. 5, May 2016, pg.163 – 168

Data Recovery in Permanent Failure of Public Cloud

Ms. Nandini.K[1], Ms. Sridevi.S[2]

[1]M.Tech Student Department of computer Science and Engineering, New Horizon College of Engineering, Bangalore, India
[2]Assistant Professor, Department of Computer Science and Engineering, New Horizon College of Engineering, Bangalore, India
Knandini804@gmail.com [1]; ssdevikumar@gmail.com [2]

ABSTRACT:- Cloud computing is gaining extensive popularity because of its intrinsic resource-sharing, flexible storage capacity. To provide fault tolerance for cloud storage, recent studies propose to stripe data across multiple cloud vendors. Public cloud includes IT resources that are delivered as service. However, if a cloud suffers from permanent failure and loses all its data, we need to repair the lost data with the help of the other surviving clouds to preserve data redundancy. We present a less repair traffic and less monetary cost to data transfer and data recovery method is called Functional minimum storage regenerator (FMSR).We validate that FMSR codes provide significant monetary cost saving in repair over RAID-6 codes, While having comparable response time performance in normal cloud storage operations such as upload/download.

Keywords:-RAID-6, FMSR, MSD, Redundancy, Fault-tolerance, Parity Bit

I. INTRODUCTION

With the rapid growth of data production in companies, the requirement of storage space grows very largely as well. Cloud storage is a concept which is an extension and development from cloud computing. Cloud storage provides effective solutions for network mass data storage. Cloud storage, in which data is striped over multiple storage nodes in a networked environment. For data availability, a storage system must keep user data for a long period of time and allow user to access their data on demand.

One way to ensure data availability is to store redundant data over multiple storage nodes. Redundancy can be achieved via maximum distance separable (MDS) code. This work focuses on unexpected permanent cloud failures. Permanent failure occurred based on malicious attacks, accidently lose data and disasters. It is necessary to activate repair to maintain data redundancy and fault tolerance. A repair operation retrieves data from existing surviving cloud over the network and reconstructs the lost data in a new cloud.

© 2016, IJCSMC All Rights Reserved

II. PROBLEM STATEMENT

Permanent failure is a long term failure, where data will become unavailable permanently. There are several situations like data-centre outage in disaster, malicious attacks where permanent cloud failure occurs and also cloud had lost accidently. However usage of multi-cloud storage leads to multi point of failure. Solution is to stripe data across multi-node storage using RAID-6 concept. Regeneration of lost data from storage nodes over the network and will reconstructs the lost data to minimize the repair traffic over network by using FMSR.

The file upload by cloud user is divided into four native chunks. Each code chunk is encoded and forms a coded chunk. Then Encoded chunk stored in multi-node by using linear combinations of original data, called parity chunks. All multi nodes are interconnected to each other.

When Recovering failed data from public cloud using functional minimum storage regenerating [FMSR] codes the whole encoded file must be decoded first if parts of a file are accessed. Two-phase checking scheme, which ensures that double- fault tolerance is maintained in the current and next round of repair.

III. RELATED WORK

The simplest form of redundancy is replication. In this method files are divided into block and each block is striped cross 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(Redundant Array Inexpensive Disk).We define the repair traffic as the amount of outbound data being the multi-cloud failure recovery. We seek to minimize the repair traffic for cost-effective repair. Here, we do not consider the inbound traffic (i.e., the data being written to a cloud), as it is free of charge for many cloud providers.

We now study the repair traffic involved in different coding schemes via examples. Suppose that we store a file of size M on four clouds, each viewed as a logical storage node. Let us first consider conventional RAID-6 codes, which are single-fault tolerant, Here we consider a RAID-6 code implementation based on the Reed-Solomon code, as shown in Figure 1.

We divide the file into two native chunks(i.e. A and B) of size M/2 each. We add two code chunks formed by the linear combinations of the native chunks. Suppose now that Node 1 is down. Then the proxy must download the same number of chunks as the original file from two other nodes (e.g., B and A+B from Nodes 2 and 3, respectively), It then reconstructs and stores the lost chunk A on the new node. The total storage size is 2M, while the repair traffic is M.

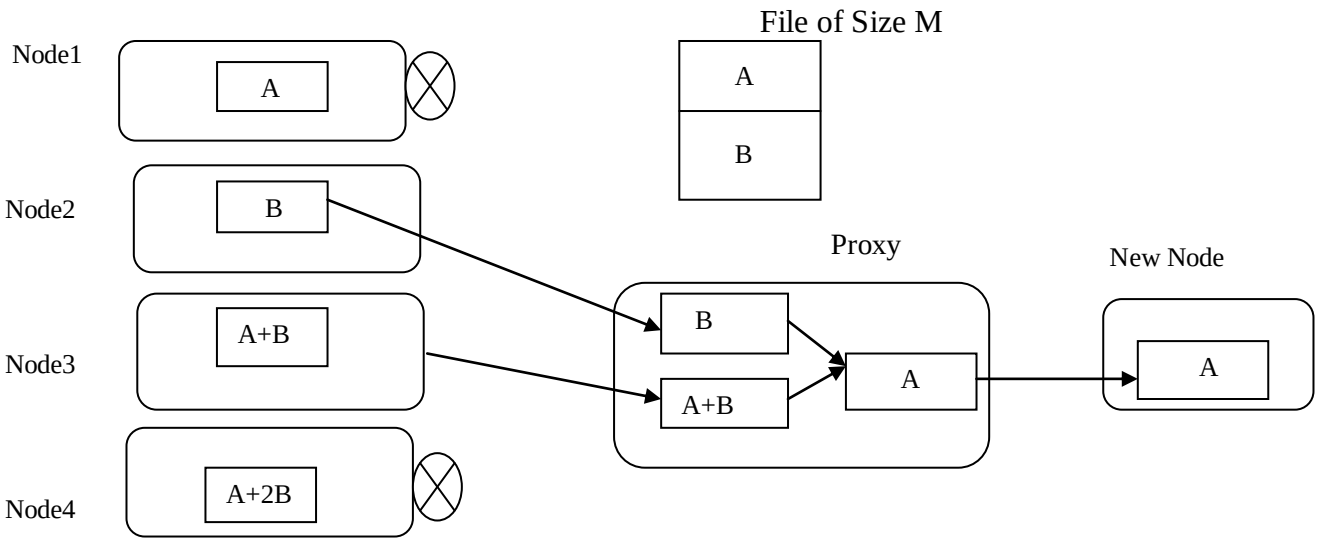


Figure 1. RAID-6 codes

© 2016, IJCSMC All Rights Reserved

Another related approach is Exact minimum-storage regenerating (EMSR), Regenerating codes have been proposed to reduce the repair traffic. One class of regenerating code is called the EMSR codes. EMSR codes keep the same storage size as in RAID-6 codes, While having the storage nodes send encoded chunks, and allocate the native and code chunks as shown in figure. Suppose node 1 is down. To repair it, each surviving node sends the XOR summation of the data chunks to the proxy, Which then reconstructs the lost chunks. EMSR codes leverage the notion of network coding, as the nodes generate encoded chunks during repair.

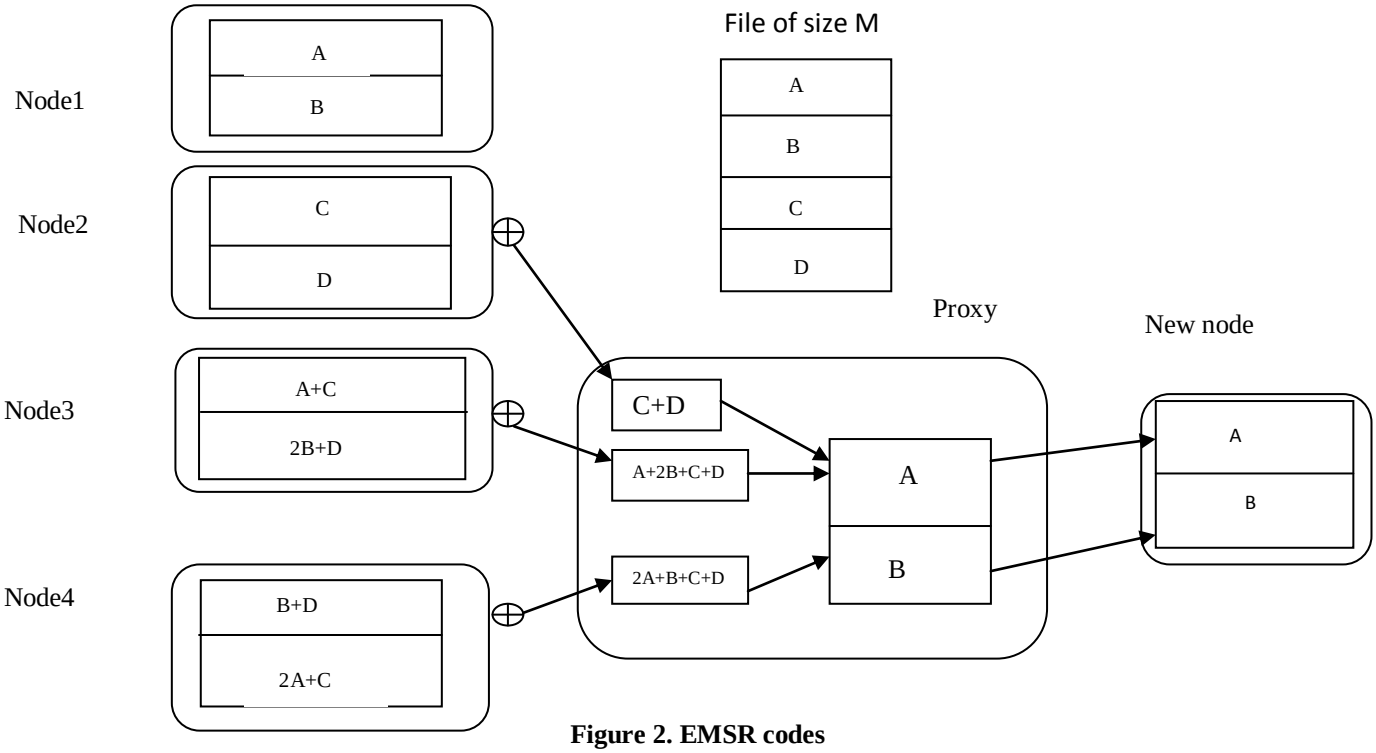


Figure 2. EMSR codes

IV. MOTIVATION

Unlike transient failure where the cloud is assumed to be able to return to normal, permanent failure will make the hosted data in the filed cloud no longer accessible, so we must repair and reconstruct the lost data in a different cloud or storage site in order to maintain the required degree of fault tolerance. In our definition of repair, we mean to retrieve data only from the other surviving clouds, and reconstruct the data in a new cloud or another storage site.

In this Section, we discuss the importance of repair in cloud storage, especially in disastrous cloud failures that make stored data permanently unrecoverable. We consider two type of failure: transient failure and permanent failure.

Transient failure:- A transient failure is expected to be short-term, such that the "failed" cloud will return to normal after some time and no outsourced data is lost. We thus expect that transient failure are common, but they will eventually be recovered. If we deploy multi-cloud storage with enough redundancy, then we can retrieve data from the other surviving clouds during the failure period.

Permanent failure:- A permanent failure is long-term, in the sense that the outsourced data on a failed cloud will become permanently unavailable. Clearly, a permanent failure is more disastrous than a transient one. Although we expect that a permanent failure is unlikely to happen, there are several situations where permanent cloud failure are still possible.

- **Data center outages in disasters.** AFCOM found that many data center are ill-prepared for disasters. For example, 50% of the respondents have no plans to repair damages after a disaster. It was reported that the earthquake and tsunami in north-eastern Japan in March 11, 2011 knocked out several data centers there.
- **Data loss and corruption.** There are real-life cases where a cloud may accidentally lose data. In the case of Magnolia, half a terabyte of data, including its backups, are all lost and unrecoverable.
- **Malicious attacks.** To provide security guarantees for outsourced data, one solution is to have the client application encrypt the data before putting the data on the cloud. On the other hand, if the outsourced data is corrupted, then even though the content of the data is encrypted and remains

© 2016, IJCSMC All Rights Reserved

confidential to outsiders, the data itself is no longer useful. AFCOM found that about 65 percent of data centers have no plan or procedure to deal with cyber-criminals.

Parity bit or check bit. Bit ia a bit added to the end of a string of binary code that indicates whether the number of bits in the string with the value one is even or add. Parity bits are used as the simplest form of error detecting code.There is two variants of parity bits: **even parity bit and odd parity bit**.

In the case of even parity, for a given set of bits, the occurrence of bits whose value is 1 is counted. If that count is odd, the parity bit value is set to 1, making the total count of occurrences of 1's in the whole set (including the parity bit) an even number. If the count of 1's in a given set of bits is already even, the parity bits value remains 0.

In the case of odd parity, the situation is reversed. For a given set of bits, if the count of bits with a value of 1 is even, the parity bit value is set to 1 making the total count of 1's in the whole set (including the parity bit) an odd number. If the count of bits with a value of 1 is odd, the count is already odd so the parity bits value remains 0.

A parity bit, or check bit is a bit added to the end of a string of binary code that indicates whether the number parity data is used by some RAID levels to achieve redundancy. If a drive in the array fails, remaining data on the other drives can be combined with the parity data (using the Boolean XOR function) to reconstruct the missing data.

For example, suppose two drives in a three-drive RAID 5 array contained the following data:

Drive 1:01101101

Drive 2:11010100

To calculate parity data for the two drives, an XOR is performed on their data:

Drive 1:01101101

Drive 2:11010100 XOR

Drive 3 10111001

The resulting parity data, 10111001, is then stored on drive 3.Should any of the three drives fail, the contents of the failed drive can be reconstructed on a replacement drive by subjecting the data from the remaining drives to the same XOR operation. If drive 2 were to fail, its data cloud is rebuilt using the XOR results of the contents of the two remaining drives, Drive 1and Drive 3:

Drive 1:01101101

Drive 3:10111001

As follows:

Drive 3:10111001

Drive 1:01101101 XOR

Drive 2:11010100

The result of that XOR calculation yields Drive 2 contents,

V. PROPOSED SYSTEM

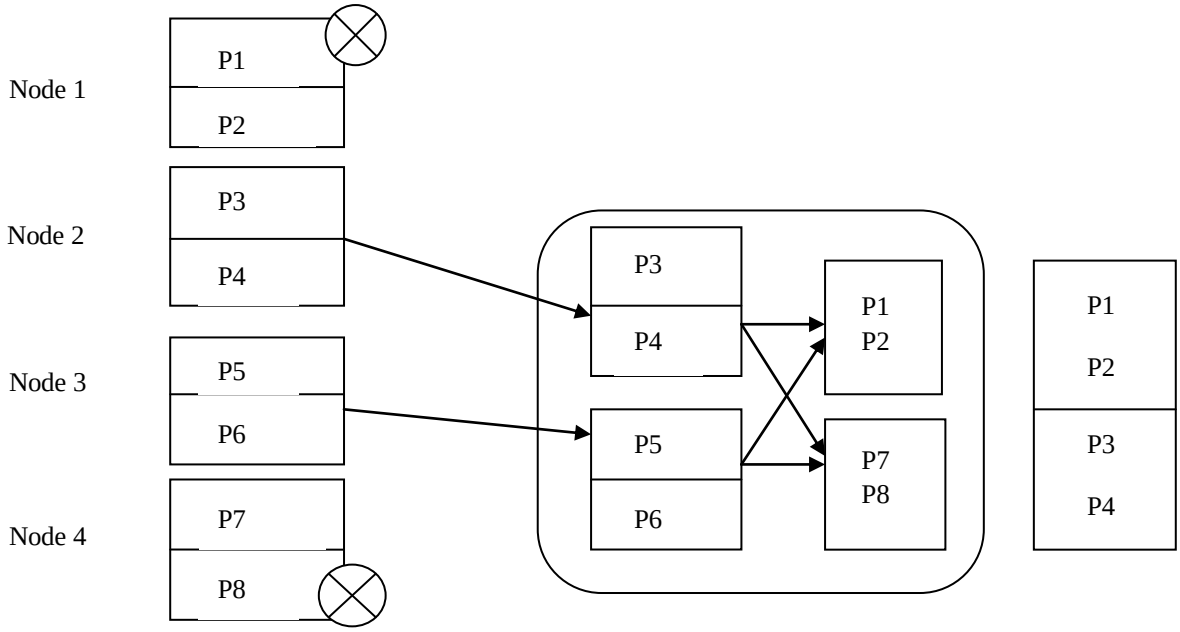
In this project, we present the design and implementation of proxy based storage system designed for providing fault-tolerant storage over multiple cloud storage providers. The first implementable design for the functional minimum-storage regenerating (FMSR) codes. Our FMSR code implementation maintains double-fault tolerance and has the same storage cost as in traditional erasure coding schemes based on RAID-6 codes, but uses less repair traffic when recovering a single-cloud failure.

While this work is motivated by and established with multiple-cloud storage in mind, we point out that FMSR codes can also find application in general distributed storage system where storage nodes are prone to failure and network transmission bandwidth is limited. In this case, minimizing repair traffic is important for reducing the overall repair time. We now present the details for implementing FMSR codes in multiple-cloud storage.

© 2016, IJCSMC All Rights Reserved

File upload by cloud user is divided into four native chunks. Each code chunk is encoded and forms a coded chunk. Then encoded chunk storing in multi-node by using linear combination of original data called parity chunks. All multi nodes interconnected to each.

When Recovering failed data from public cloud using functional minimum storage regenerating (FMSR) codes the whole encoded file must be decoded first if parts of a file are accessed. Two-phase checking scheme, which ensures that double-fault tolerance, is maintained in the current and next round of repair.



We propose a two-phase checking scheme, which ensures that the code chunks on all nodes always satisfy the MDS property, and hence data availability, even after iterative repairs. In this section, we analyze the importance of the two-phase checking scheme.

VI. IMPLEMENTATION

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

User: This module consists registration, login, and group-creation, join request and accept request operations. User first register by giving input username, password and email. Once register they can update his details after login.

They should create a group before upload data. Other users can register under this group. Once user confirmed other user can access the upload 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 operation.

1. Ranch code chunks is encoded and forms a coded chunk. Code chunk is denoted as P.

Each P_i is formed by linear combination of the 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 download 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.

© 2016, IJCSMC All Rights Reserved

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.

The repair traffic as the amount of outbound data being downloaded from the other surviving clouds during the multi-cloud failure recovery. To minimize the repair traffic for cost-effective repair. To generalize double-fault-tolerant MDS codes for n storage nodes, divide a file of size M into 2(n-2) native chunks, and use them to generate 2n code chunks. In contrast, for RAID-6 codes, the total storage size is also Mn/n-2, while the repair traffic is M. Our proposed systems reduce repair traffic by 50% of repair traffic.

VII. CONCLUSIONS

In this paper proposed a multi node failure recovery using network coding (RAID6) method. We used a proxy-based, multiple-cloud storage system that practically addresses the reliability of today's cloud backup storage. This paper not only provides fault tolerance in storage, but also allows cost-effective repair when a cloud permanently fails. Recovery from multimode failure takes place only if the following condition is met: M=N-2 (N: number of nodes & M: number of failure nodes).

Download the same amount of original data by connecting to any k nodes. While in traditional RAID-6 codes, the original amount of data is retrieved to recover the lost data. Thus, traditional RAID-6 codes retrieve the same amount of data in degraded reads, while RAID-6 codes have higher computational overhead in decoding.

References

[1] H. Abu-libdeh, L. Princehouse, and H.Weatherspoon. RACS: A Case for Cloud Storage Diversity. In Proc. Of ACM Socc,2010.
[2] Amazon Web Services, AWS Case Study:Backupify, http://aws.amazon.com/solution/case-studeies/backupify/, 2013.
[3] M. Arrington, "Gmail Disaster: Reports of Mass Email Deletions," Dec.2006. http://www.techcrunch.com/2006/12/28/g mail-disaster-report-of ass-email deletions.
[4] Liu Hao, Dezhi Han, "The study and design on secure-cloud storage system", In IEEE society, 2011.
[5] Rashmi, Nihar B.Shah, and P.vijay Kumar, fellow, "IEEE Optimal exact regenerating Storage at the MSR and MBR Points Via a Product-Matrix Construction" VOL.57,NO. 8, AUGUST 2011.

© 2016, IJCSMC All Rights Reserved