A Novel Technique for Storage in Cloud Data Centers

Makhan Singh
Computer Science and Engineering, University Institute of Engineering & Technology, Panjab University, Chandigarh, India.
singhmakhan@pu.ac.in

Abstract—An Information Dispersal Algorithm takes a file F as a input and divide this file into n pieces that needs to be dispersed among n nodes such that any m pieces out of total n pieces will be sufficient to reconstruct the whole file F. The size of each piece is |F| / m. It must also ensure that the complete knowledge of any m-1 pieces is insufficient to reconstruct the complete file F. The ideas for accomplishing this have been given in many literatures in the past. This manuscript discusses the application of Information Dispersal Algorithms in Storage of files on cloud Data Centers instead of using previous schemes of replication.

Keywords- *Information Dispersal Algorithm, cloud computing, backup, data centers*

*****

I. Introduction

Information Dispersal is the ability to break a piece of information into a desirable number of pieces such that the loss of a threshold number of these pieces does not mean the loss of the complete information. Information Dispersal Algorithms are cryptographic schemes used to generate multiple files, or shares of data, from a given file. Redundancy is the major benefit, since more than one predetermined subset of shares can be used to reconstruct the file. A threshold subset of shares is needed to later reconstruct the file. Below this threshold the original file cannot be reconstructed, even partially. Dispersing shares of data geographically can achieve extraordinary protection against loss as well as information security. In IDAs a piece of file F needs to be dispersed among n nodes such that any m pieces will be sufficient to reconstruct the whole file F. The size of each piece is |F/m|. We must also ensure that the complete knowledge of any m-1 pieces is insufficient to reconstruct the complete file F. Information dispersal algorithms have different applications, as secure and reliable information storage, fault-tolerant and efficient transmission of information in distributed systems, and communications between processors in parallel computers [1].

Cloud computing is a networked environment where data is stored in various remote servers so as to be used by many different users through online access. The cloud computing architecture is diagrammatically shown below:

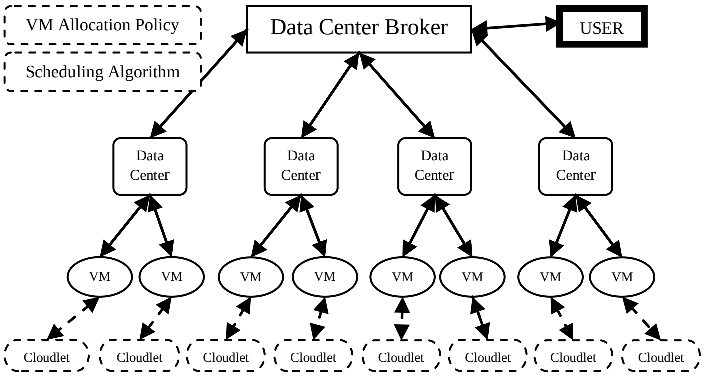

Figure 1. Architecture of Cloud Computing Environment

The figure 1shows the architecture followed by Cloud Computing that will be used in this manuscript. Here the Data Center Broker has multiple Data Centers associated with it. It also has a VM allocation policy that associates 1 or multiple VMs to each Data Center. The VMs each have cloudlets attached to it. The Data Center Broker also has a scheduling policy associated with it that schedules each VM and cloudlet.

Backup of data and cloud storage services are an important feature for which cloud data centers are being popularly used by businesses seeking affordable alternatives to off-site data storage and backup. However, evaluating vendors is a difficult task and many areas like data security, service infrastructure and performance should be considered to make sure that data is protected by a service provider. The reliability and availability of the service is an equally important consideration because cloud services will really only be useful if they are both secure and accessible when most needed. Encryption of data is often done at cloud data centers to make it more secure and less prone to theft. We also often need to keep a backup of the files that the users entrust the cloud service providers with so as to minimize the loss of data in case some of the data on the cloud data centers is lost due to potential hardware malfunctions. The data on cloud data centers is kept in a redundant environment which creates ‘mirrors’ or copies of all the data on the cloud data centers do as to increase redundancy and availability of the system. But this approach to keeping backup of data is inefficient due to large storage overheads and security issues. Also if the ‘mirrors’ or copies of the data were lost this would mean the loss of the whole data and information. So to make backup of data on cloud data centers more efficient, we can use Information Dispersal Algorithms to decompose the data into number of parts and distribute these parts of data on various data nodes and data can be reconstructed from any threshold number of these parts. This reduces the storage overhead considerably and also makes the data easily available even if some of these nodes are down. Also any number of parts of the file less than the threshold number give no significant information to the user.

Not much effort has been put in this field of creating backup of data on cloud data centers using Information Dispersal Algorithms. Replication of data was considered the only option and this increased storage overheads and also reduced security of the data. Thus Information Dispersal

Algorithms can offer considerable improvements over the previous backup techniques which employed replication for backup.

II. Related Work

Adi Shamir [2] proposed an Information dispersal Algorithm in 1979 keeping in mind the problem of secret key sharing. He used the concept of polynomial interpolation to divide the information at hand into the desired number of pieces. This however had a drawback that all the parts of the data were of the same size as the data itself and thus it led to a lot of storage overhead. G.R. Blakley [3] (1979) also approaches the problem of information dispersal keeping safeguarding cryptographic keys in mind. It uses the concept of vectors to divide and distribute the key or data. Micheal O. Rabin [4] proposed an algorithm for 'Efficient Dispersal of Information for Security, Load Balancing and Fault Tolerance' in 1989. He proposed a method for reliably dispersing a file F into n pieces such that it can be reconstituted from any m pieces. His method can be viewed as belonging to the field of error correction codes [5], in which extra bits are added to a message creating a block, so that after occurrence of any k errors within the block, the message can still be reconstructed. K. Kurosawa, W. Ogata and S. Tsujii [6] in 1993, proposed an algorithm that discusses secret sharing and employees IDA in sharing of secret data. Hugo Krawczyk [7] also proposed an information dispersal algorithm in 1994 while making an attempt to make secret sharing short. In his scheme the secret is first encrypted using a key K, then it is partitioned using an Information Dispersal Algorithm, then this share is privately transmitted to various participants via the network. The recombination of files is achieved using the reverse of the above scheme. Philippe Beguin, Antonella Cresti [8] in 1998 analyzed the distribution a file among a set of users such that a threshold number of these users can reconstitute the file. Their scheme is an extension of scheme proposed by M.O. Rabin [4] and it determines a bound on the amount of information made available to a user and the application of this scheme in cryptography. Alfredo De Santis and Barbara Masucci [9] discussed the various Information Dispersal Algorithms in 2002 by using an information- theoretic framework. They compared algorithms by discussing their technique for splitting and recombining the file. Also they discussed the mathematical concepts in each of these algorithms.

Various applications of the Information Dispersal Algorithms have been discussed and documented by various authors. Hung-Min Sun and Shiuh-Pyng Shieh in [10] discuss the application of IDA in the field of Reliable Communication in Computer Networks. The application of IDA in Dispersal of probabilistic Latency Targets is discussed by Marvin K. Nakayama and Bulent Yener in [11]. Azar Bestavros in [12] discusses IDA with respect to Time –Critical Reliable Communication. The applications with respect to Combinatorics, Compression, Security and Transmission have been discussed by M. O. Rabin in [13].

Giampaolo Bella, Costantino Pistagna, and Salvaore Riccobene [14] discussed the use of Information Dispersal Algorithms in efficient distributed backup of data in 2006. They discuss the application of Information Dispersal Algorithms in the backup of file on a flat, non-hierarchical network which is implemented using the Chord location service. The previous techniques for the backup of data on cloud data centers have been discussed in many manuscripts. Timothy Wood, Emmanuel Cecchet, K.K. Ramakrishnan, Prashant Shenoy, Jacobus van der Merwe, and Arun Venkataramani [15] discuss the mechanisms of backup of data on cloud computing environment with respect to its application disaster management of the files on the cloud network. In [16], the Hitachi Data Systems discuss the tape based backup of data and information reduction with respect to their cloud environment. The Google File System [17] also discusses the techniques for the backup of data on the google cloud environment. However these methods for the backup of data are neither well formed nor well discussed. Most of the methods are localized to particular cloud environments and have not been given much priority while designing the cloud structure.

Therefore in this paper Information Dispersal Algorithm proposed by Micheal O. Rabin based scheme will be discussed in order to keep backup of sensitive files in Data Centers in any cloud environment.

III. Splitting Files In Various Data Centers

In this scheme of data backup the space efficiency and reliability of the system can be maintained. For example user has a file F that is of significant importance to the user and thus the user wants to keep backup of this file so as to not lose the file under any circumstances. But for the cloud environment to remain efficient, the backup must be as efficient as possible. Thus a scheme is proposed where Information Dispersal Algorithms is used to split the file into n number of equal parts such that the file is reconstructed back from any m of these pieces where m is less then n and all these n parts are distributed over the network to any n different data centers that each have different virtual Machines (VMs) associated with them. This scheme requires a threshold m available number of any of these pieces to reconstruct the file. So even if the other (n – m) data centers lost their data or were unavailable at the moment, the file would still be recoverable. Also the IDA discussed by Micheal O. Rabin used in scheme ensures that if any malicious user gets a hold of any m-1 of these pieces the file F would still be secure and not recoverable.

The size of each piece or fragment to be distributed is |F|/m. and |F| is the number of character in file F hence the number of characters is (n/m)*|F| and we can optimize this scheme by choosing n such that the ratio n/m is as close to 1 as possible. This scheme is much more efficient than replicating the whole file on multiple Data Centers. There are a lot of benefits that this scheme provides that we shall discuss later in this paper.

Thus we propose an algorithm to split and distribute the file as follows:

```
begin
Step-1: Select the threshold number m. This is the number
        of Data Centers that are always available.
Step-2: Get the file F from the user through the Data Center
        Broker. The following steps are followed by the
        Data Center Brokers Storage Class.
Step-3: Choose n vectors $a_i=(a_{i1},\ldots,a_{im})$ such that each
        subset of m of these vectors is linearly independent.
        The file F is a string of characters
        $F=(b_1,\ldots,b_m)(b_{m+1},..,b_{2m})$.. Denote $S_i = (b_1,..,$
        $b_m)$,    etc. For $i = 1, \ldots, n$.
Step-4: Each piece $F_i=c_{i1},c_{i1},\ldots,c_{iN/m}$ where
        $C_{ik} = a_i . S_k = a_{i1} + b_{(k-I)m+1} + \ldots.. + a_{im} + b_{km}$.
       This can be represented using metrics:
        Let A be the m*m matrix $A=(a_{ij})_{1<=i,j<=m}$ whose
        $i^{th}$ row is $a_i$.
```

$$A.\begin{matrix} b1 \\ \vdots \\ bm \end{matrix} = \begin{matrix} ci1 \\ \vdots \\ cm1 \end{matrix} \Big| \qquad (1)$$

Thus we can gather each piece F_i and split the file into n number of parts.

Step-5:Distribute each of these parts to the required Data Center from the Data Center Broker.

Step-6:The Data Centers distribute these fragments to their associated VMs which each have a cloudlet attached to them.

end

This procedure can be followed to split and distribute the file to various Data Centers. The distribution of each piece to Data Center is shown in the figure given below. The user provides the file F and this file is then broken into n number of pieces F_1 to F_n. These pieces are then distributed to various Data Centers connected to the Data Center Broker. The threshold number of Data Centers m that provide for the threshold number of pieces for recovery of the file is selected based on the fact that these minimum number of Data Centers must always be available during recovery.

The figure 2 below explains the whole splitting and distribution procedure for the file F for the backup on various Data Canters.

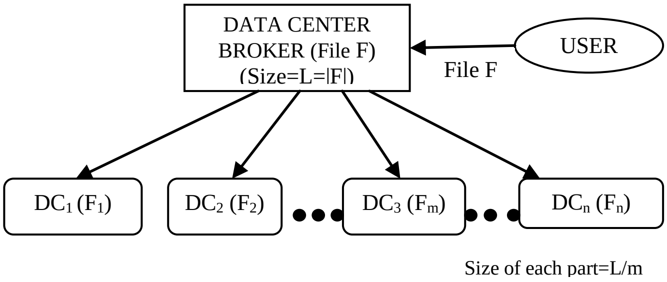

Figure 2. Splitting and Distribution of file F to various Data Centers

IV. Recombining the File

The recombination procedure for the file F is depicted in the figure 3 given below. In this figure the file F is reconstituted from any m pieces available from F_1,.., F_n. The m Data Centers available at the given time provide these required m parts. We collect various parts of the file from the m available Data Centers.

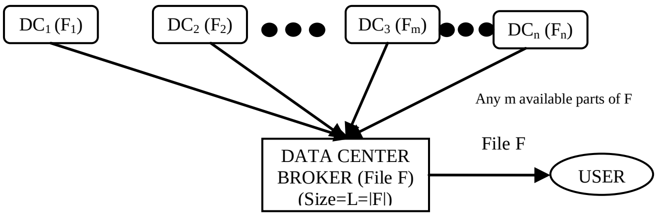
DC₁ (F₁)
DC₂ (F₂)
DC₃ (Fₘ)
DCₙ (Fₙ)
Any m available parts of F
File F
DATA CENTER BROKER (File F) (Size=L=|F|)
USER

Figure 3. Recombining file F from any m Data Centers

We propose the following algorithm to reconstitute the file F from the various fragments or pieces collected from the m available Data Centers at the instant when the file is recollected.

begin

Step-1: Collect the m threshold number of pieces from m available Data Centers.

Step-2: The file can be recovered from the matrix representation given below

$$\begin{matrix} b1 \\ \vdots \\ bm \end{matrix} = A^{-1}.\begin{matrix} ci1 \\ \vdots \\ cm1 \end{matrix} \qquad (2)$$

Step-3: The file F=(b_1,…,b_m)(b_{m+1},..,b_{2m}). So we can use the above equation for recovering the file and forward it to the user.

end

Since the above matrix requires a minimum of m coefficients the file is not recoverable even if m-1 parts are intercepted by malicious user.

V. Benefits of the Proposed Scheme

This scheme provides the following benefits for security concerns related to sensitive file on Data Centers:

- Reliability: The reliability of the whole cloud environment increases because the sensitive files can be recovered even if only a threshold number of its parts are available. So even if a few data centers were down or the data on these data centers was lost the file could still be recovered a reconstituted.
- Availability of service: Availability for the whole system depends on the fact that the file should be available even when a few data centers are down or inaccessible. The information dispersal scheme provides for this requirement so therefore the availability of data automatically increases.
- Optimized redundancy of data: Using the information dispersal scheme we can backup the data in sensitive files without having to create many replicas which was originally used for keeping the backup of files in cloud computing environment. This reduces redundancy of data.
- Security of backup: The security of the files present in the cloud data centers is also an important issue for cloud service providers as well as users. The information dispersal method ensures the security of the files because the given file cannot be reconstituted if we have even 1 less than the threshold number of parts required for the file.
- Efficient management of resources: The information dispersal scheme provides for efficient management of resources on various cloud data centers because all the data centers have equal parts of the file F whereas in replication the whole file F is stored in various locations multiple times. Thus this scheme is more efficient.

These can become the factors that determine if this scheme is better than replicating the entire file over multiple Data Centers.

A. Comparison Metrics

The metrics for comparing the proposed scheme are as follows:

- Storage/Size of shares: The comparison of the storage requirements of the replication and the scheme proposed are done on the basis of the total storage space used which depends on the size of each share distributed and that is given by $|F_i|=|F|.(n/m)$ for the above discussed scheme. Therefore storage $S_i=n.|F_i|=|F|/m$

- Availability: The availability of the file is given by the ratio of the percentage of data centers that can be inaccessible at the particular instant and the total redundant space acquired by the file over all the data centers. This ratio can mathematically be represented as follows:
  Availability A= (n-m/n)/total redundant data
- Redundancy: The redundancy is the percentage of redundant data present in the system that is how many times each byte of information is repeated. Thus it can be calculated as follows:
  Redundancy R = [(total storage-size of File)/size of file]*100
  = [(S_i-|F)|/|F|]*100

The comparison of the scheme proposed in this manuscript to the existing replication scheme can be done using the above metrics and can prove that proposed scheme in this manuscript provides significantly improved performance.

B. *Simulating the Proposed Scheme*

The above scheme can be simulated using a simulator that can replicate the Cloud computing environment. The simulator that is used here is known as CloudSim. The simulator is a java based simulator that provides various classes that replicate the Cloud environment on the system. The objects of each of these classes are used to call the various methods associated with them. Above algorithm is simulated on the CloudSim environment to get experimental results and prove that the scheme proposed above provides some significant improvement on the existing techniques. The process followed to simulate the above scheme is as shown in the figure 4.

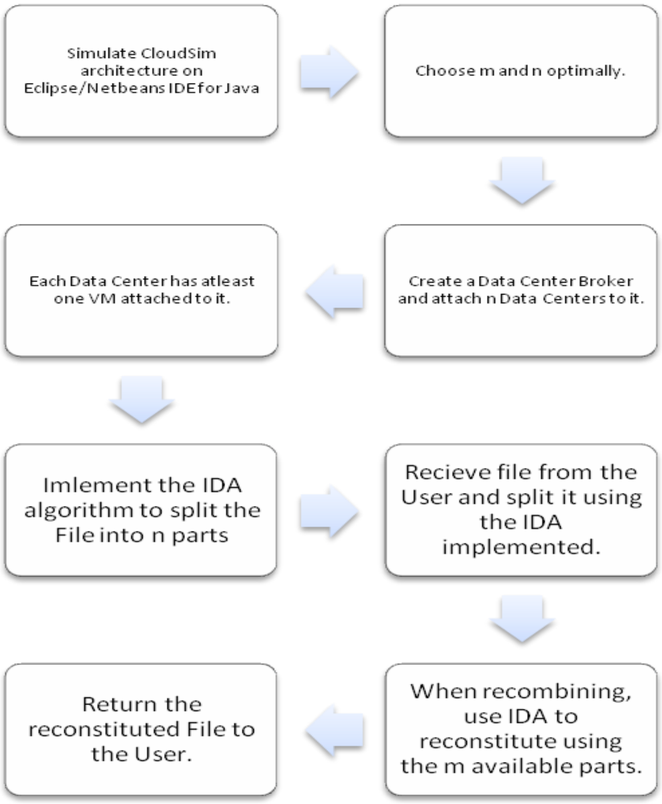
Simulate CloudSim architecture on Eclipse/Netbeans IDEforJava
Choose m and n optimally.
Create a Data Center Broker and attach n Data Centers to it.
Each Data Center has atleast one VM attached to it.
Imlement the IDA algorithm to split the File into n parts
Recieve file from the User and split it using the IDA implemented.
When recombining, use IDA to reconstitute using the m available parts.
Return the reconstituted File to the User.

Figure 4. Simulation process for proposed scheme for Backup of files.

The simulation results observed after following the above process are as follows:

Factor of Comparison	No. of Data Centers (n)	Value of threshold (m)	Size of File F (in KBs)	Value with Replication	Value using proposed scheme
Storage/ Size of share (in KBs)	10	8	1.2	12	1.5
Reliability (P(r))	10	8	1.2	.075	.133
Redundancy (percentage)	10	8	1.2	1.2	25

The graphs to show the variation of the above metrics are given below:

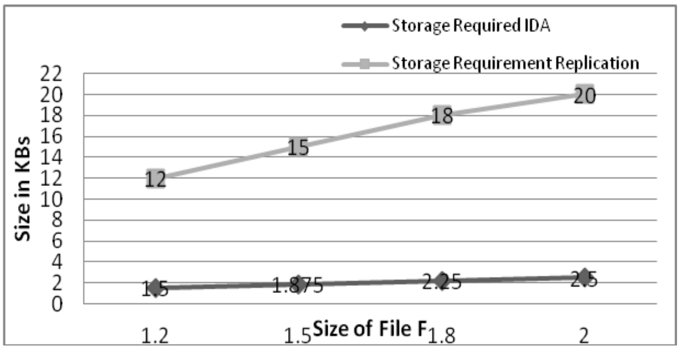

Figure 5. Graph showing variation of Storage Requirement to the size of File

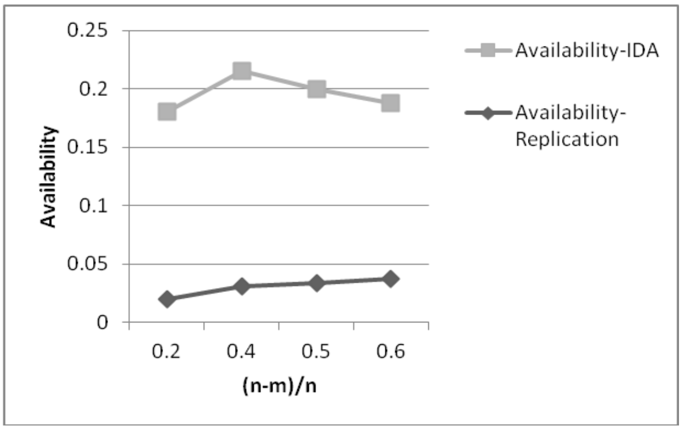

Figure 6. Graph for Availability Variation with the value of (n-m)/n

VI. CONCLUSION

The backup of files using Information Dispersal Algorithm is a scheme that provides for efficient and secure method for backup of data on Cloud Data Centers. It has many advantages over the former backup scheme which employed replication of the file on multiple Data Centers. This scheme can easily be tested using any simulator that can provide a simulated Cloud environment.

REFERENCES

[1] A.D. Santis and B. Masucci, "On Information Dispersal Algorithms", Proc. of 2002 IEEE International Symposium on Information Theory – ISIT, 2002

[2] A. Shamir, "How to share a secret", Communications of the ACM, vol-22, 11th Nov. 1979, pp. 612-613.

[3] G.R. Blakley, "Safeguarding cryptographic keys", National Computer Conference, 1979, pp. 313-317

[4] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance", Journal of the ACM, vol-36, 1989, pp. 335–348.

[5] R. Berlekampe, "Algebraic coding theory", McGraw-Hill, New York, 1968

[6] K. Kurosawa, W. Ogata, S. Tsujii, "Nonperfect Secret Sharing Schemes", Advances in Cryptology - AUSCRYPT 92, Lecture Notes in Computer Science, vol. 718, pp. 56-66, 1993.

[7] H. Krawczyk, "Secret Sharing Made Short", Advances in Cryptology, 1994, pp-136-146

[8] P. Beguin and A. Cresti, "General information dispersal algorithms", Theoretical Computer Science, vol- 209, 1998, pp-87-105

[9] A.D. Santis and B. Masucci, "On Information Dispersal Algorithms", International Symposium on Information Theory, 2002, pp-410

[10] H.M. Sun and S.P. Shieh, "Optimal Information Dispersal for Reliable Communication in Computer Networks", IEEE, pp. 460-464, 1994.

[11] M. K. Nakayama and B. Yener, "Optimal information dispersal for probabilistic latency targets", Computer Networks, vol-36, pp. 695-707, 2001

[12] A. Bestavros, "An Adaptive Information Dispersal Algorithm for Time-Critical Reliable Communication", Network Management and Control, Springer Science+Business Media, pp. 423-438, 1994

[13] M. O. Rabin, "The information dispersal algorithm and its applications, Combinatorics, Compression, Security and Transmission", Springer, 1990, pp. 406-419.

[14] G. Bell, C. Pistagna and S. Riccobene, "Distributed backup through information dispersal", Electronic Notes in Theoretical Computer` Science, vol-142, 2006, pp. 63-67

[15] Timothy Wood, Emmanuel Cecchet, K.K. Ramakrishnan, Prashant Shenoy, Jacobus van der Merwe, and Arun Venkataramani, "Disaster Recovery as a Cloud Service:Economic Benefits & Deployment Challenges", Proceedings of the 2nd USENIX coneferce on Hot Topics in Cloud Computing,2010,pp. 8-8

[16] Extend Enterprise File Services Beyond the Data Center; Mobilize Data Among Datacenters and Remote, Branch Offices, Hitachi Data Systems, 2012.

[17] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, "The Google file system", Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP '03), ACM, 2003, pp. 29-43.