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: 6.017

IJCSMC, Vol. 5, Issue. 12, December 2016, pg.86 – 91

A Review of Information Dispersal Strategies in Distributed Cloud Storage Systems

Ujjval Kumaria[1], Darshan Doshi[2]

[1, 2]Department of Information Technology, Mukesh Patel School of Technology Management and Engineering, NMIMS University, India

[1] ujjvalkumaria@gmail.com; [2] darshandoshi45@gmail.com

Abstract— In this paper we take a closer look at distributed cloud storage systems. We provide an overview of information dispersal strategies to realise reliable distributed cloud storage systems and provide an overview of state-of-the-art cloud storage approaches. Then, we analyse them with respect to security properties. Furthermore, we discuss the lack of privacy features and in particular features to provide access privacy in existing distributed cloud storage systems, which is an important direction for future research on distributed cloud storage. We also look at how the algorithms being compared are able to distribute data in accordance with node capacity and whether they can retain uniformity with minimum data movements when nodes are added and removed.

Keywords— cloud storage; distributed; replication; data integrity; redundancy

I. INTRODUCTION

Cloud storage today is one of the most prevailing choice of data storage amongst enterprises all across the world. The services often extend from solutions like archiving to that of backup, sharing of data as well as the synchronisation of data. There are quite a few obvious advantages that make cloud storage such an attractive choice amongst users and enterprises, both big and small, like the omnipresent access to data, the immediate scalability as well as the cost-effective financial implications of the billing model of cloud service providers.

Before looking at approaches and strategies for cloud storage, we need to look at cloud storage as the infrastructure of cloud computing to understand how service providers need to be flexible in their service delivery and subsequently keep users secluded from the underlying infrastructure because the demands and requirements of a user are never constant, they always keep varying therefore giving rise to the need for the aforementioned flexibility [11]. For achieving this layer of abstraction, providers make use of Virtual Machines (VM) that allow both the isolation of applications from the underlying hardware and other VMs as well as the customization of the platform to suit the needs of the end-user. Some of the popular service providers such as Amazon, Google, Salesforce, IBM, Microsoft, and Sun Microsystems have begun to establish new data centres for hosting Cloud computing applications in various locations around the world to provide redundancy and ensure reliability in case of site failures.

The advantages of cloud storage, both, financially and in terms of security are conspicuous. In terms of the fiscal matters, the usage of virtual resources exponentially reduce overall costs for users as compared to physical resources. As far as security is concerned, the data stored on the cloud gets duplicated across multiple physical machines making it safe from accidental erasure or hardware crashes, so that the cloud continues to function normally even if one or machines go down.

Despite the clear advantages, there are still a number of potentially series issues pertinent to cloud storage which are currently hindering the widespread adoption of storage clouds among enterprises [6]. These concerns are mainly devoted to missing or inadequate security and privacy related features, requiring customers to fully trust in the integrity of the cloud provider as well as the provider's security practices.

Another common issue is being dependent on a single provider which is a catalyst for some critically underlying problems. First, it can cause the so-called vendor lock-in problem for the customers, which results in prohibitively high cost for clients to switch from one provider to another. Second, it can cause service disruptions, which in turn will lead to SLA violation, due to cloud outages, resulting in penalties, monetary or other forms, for the service providers. Examples of outages and data loss incidents of noteworthy cloud storage services have appeared from time to time. Third, the users may not retain their own copy of the data once it has been stored or outsourced to storage cloud giving the service provider motive to be unethical, for example, by data that is rarely accessed data without being detected. Similarly, the provider may even attempt to hide data loss incidents in order to maintain their reputation [8].

A possible solution to these problems is by using multiple, independent cloud providers, also known as Cloud-of-clouds approach, which is somewhat a new concept in the industry which will be explored in the paper and has so far proved to be an effective way to provide better availability for the cloud storage systems. Such approaches reduce the risk of being influenced from single cloud provider outages and provide (at least to some degree) resistance to loss or corruption of data at cloud providers [3].

In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. Outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, however its lack of offering of strong assurance of data integrity and availability may impede its wide adoption by both enterprise and individual cloud users.

Within the last few years several approaches to eliminate deficiencies within state of the art cloud storage systems have been proposed. In this paper, we provide an overview of existing approaches and evaluate them.

II. STORAGE CLOUD INFRASTRUCUTRE/ARCHITECTURE

When storing data on the cloud, the user sees a virtual server—that is, it appears as if the data is stored in a particular place with a specific name. Commonly known as Storage as a Service (StaaS), it facilitates cloud applications to scale beyond their limited servers. StaaS allows users to store their data at remote disks and access them anytime from any place. Cloud storage systems are expected to meet several rigorous requirements for maintaining users' data and information, including high availability, reliability, performance, replication and data consistency.

Another way to define cloud storage is: Once they've gathered enough stuff, they have to find places to store all of it. Cloud storage has several advantages over traditional data storage [2]. For example, if you store your data on a cloud storage system, you'll be able to get to that data from any location that has Internet access. We won't need to carry around a physical storage device or use the same computer to save and get back our information. With the right storage system, you could even allow other people to access the data, turning a personal project into a collaborative effort.

Cloud Storage Reference Model

It is a model created which shows multiple types of cloud data storage interfaces able to support both legacy and new applications. All of the interfaces allow storage to be provided on demand, drawn from a pool of resources. CDMI or the Cloud Storage Management Interface is the interface used to create, delete, retrieve, and update all the information on the cloud. Basically with the help of this interface the client will be able to understand the capabilities of the cloud and he could manage the cloud with it [5].

This interface is also used by administrative and management applications to manage containers, accounts, security access and monitoring/billing information, even for storage that is accessible by other protocols.

Here are five key benefits of using cloud storage and of applications that take advantage of storage in the cloud.

- Cost effectiveness
- Ease of management
- Disaster Preparedness
- Lower impact outages and upgrades
- Simplified Planning

III. LITERATURE SURVEYS

A. Improving Storage Availability in Cloud-of-Clouds with Hybrid Redundant Data Distribution

In an intercloud storage environment, data redundancy schemes are introduced to achieve storage availability, performance, and cost and space efficiency. Two standard methods of achieving redundancy.

a) Replication – This allows for data to be spread across multiple active, cloud-based, sites. Increases durability and availability. Issues –
(i) For large systems, number of replicas need to be more to have higher availability.
(ii) Increased number of replicas introduces the extra bandwidth and storage overhead to the system.
For small files, replication is effective because of small bandwidth and storage overhead.

b) Erasure code – Data is stored across a set of different providers or media after it is broken into fragments, expanded and then encoded with redundant data pieces. Provides high space efficiency and decent space availability. Issues –
(i) Extra time required to record redundancy information for small files.
(ii) Large amount of network traffic required to reconstruct data when a cloud provider suffers an outage.
(iii)Work well for large files as, access latency is less because of parallel access via multiple providers. [9]

In this paper, the authors have proposed a hybrid of the aforementioned approaches to address the availability problem of cloud storage systems called Hybrid Redundant Data Distribution (HyRD).
In HyRD, large files are distributed among multiple providers with erasure-coding while small files are replicated on multiple high-performance providers. By exploiting the best of both schemes, HyRD retains the advantages of both the replication and erasure-coding schemes.

Aims of HyRD –

1. Improving the cloud storage availability
2. Reducing the user access latency
3. Improving the cost efficiency

Architecture of HyRD –
HyRD resides on the client side and interacts with the cloud storage providers via their standard interfaces thus making it easily applicable to the service of any provider. Three main functional modules –

1. Workload monitor – classifies incoming data into small, large or metadata files based on access latency.
2. Request dispatcher – Decides which scheme should be used for incoming data, and distributes it to corresponding cloud storage providers.
3. Cost & Performance evaluator – Evaluates storage service in terms of performance and cost.

Implementation of HyRD –
Middleware of General Cloud Storage API (GCS-API) to interact with various providers. GCS-API hides complexity of CSPs, making it easy to add new CSPs to the HyRD system. Supports five functions – (a)LIST, (b)GET, (c)CREATE, (d)PUT, (e)REMOVE.
For using cloud storage service, HyRD uses RESTful APIs for explicitly taking advantage of HTTP methodologies. Evaluation module directly interacts with individual CSPs to evaluate corresponding values.

B. A Load Balancing Algorithm for Private Cloud Storage

A Cloud Storage is a repository in which the data is stored, managed and maintained which can be accessed all over the internet. There are four ways in which the data can be deployed over the cloud public, private, community and hybrid. Private Cloud is used for the specific organization exclusively. It can be accessed from anywhere but only within the organization. Public cloud is provided for open use by general public. Community Cloud is used for a group of organizations [2]. Hybrid cloud composes of two or more distinct clouds such as private, public or community clouds. There may be an increase in storage demands from the existing consumers or the number of consumers itself may increase. These situations make dynamic expansion of cloud storage with additional storage nodes a necessity.
During such expansion of the storage cluster or due to the addition and deletion of files, load imbalance may arise in the cluster. In order to maintain the balance in the cluster, a load balancing algorithm is used. This algorithm attempts to balance the load during the data placement as well as in any later situations that lead to load imbalance.

Load Imbalance:
Whenever the chunks are placed initially across the storage nodes, it is always necessary to place them in such a way that the load is balanced among the storage nodes. Subsequent usage of the storage cluster leads to load imbalance in the cluster.
This load imbalance may occur due to the following reasons:
Inclusion of a new storage node - A newly added storage node may have lesser or no data compared to the other nodes.

- Addition or deletion of files - It leads to imbalance in the cluster.
- Heavily loaded node - Whenever the load of Storage node exceeds a pre de-fined threshold, the storage node turns into a heavily loaded node.

The Algorithm Used for load balancing is:

- Coordinator Algorithm
- Data Placement Algorithm
- Load rebalance Algorithm
- Data Migration Algorithm

Load Rebalance Algorithm:
This Algorithm is responsible for periodically checking if the storage nodes are lightly or heavily loaded. In order to detect this, the ideal load for each storage node has to be found. From the ideal load, lower and higher thresholds for each storage node is found. This utilized space of each storage node is now compared with the lower and the higher threshold values. If the utilized space of the storage node is now less than that of the lower threshold, then it is known as a light weighted storage node and its status is indicated as ”low”. Similarly, if a utilized space of a storage node is greater than the higher threshold, it is considered to be as a heavy weighted storage node and its status is set as ”high” [2]. These light weighted nodes are then sorted in ascending order and the heavy weighted nodes are sorted in descending order. Once the ordering is finished, each light weighted node is paired with a corresponding heavy node for load rebalancing. This pairing of nodes is stopped if a heavy weighted node cannot find a light weighted node.

Implementation:
There is a test bed which is created using four machines. Each of the machine has the configuration of 8GB hard disk, 4GB RAM capacity, and Intel core i5, i7 processor. Out of which One of the machines is designated as the centralized coordinator (CC). Which acts as a TCP server and FTP client. The remaining three machines are the storage nodes and they act as TCP clients and FTP servers.

C. ASURA: Scalable and Uniform Data Distribution Algorithm for Storage Clusters

The Large-scale storage cluster systems need to be managed as they have a vast amount of information denoting these combinations of data identifiers (IDs) and the corresponding nodes that store the data. Management of these data systems by using the data distribution algorithms, rather than management of the data using tables, has been proposed in recent research papers. In algorithm management, the data is distributed in accordance with a data distribution algorithm that is able of determine, on the basis of the datum ID, the node in which the required data is stored [4]. The requirements for a data distribution algorithm to be efficient are short calculation times, low memory consumption of the nodes, uniform data distribution according to the capacity of each node and the ability of the system to handle the addition or removal of nodes. This paper presents a data distribution algorithm called ASURA (Advanced Scalable and Uniform storage by Random number Algorithm), which satisfies all these above requirements.
This paper focuses on an algorithm that satisfies all the conditions which are uniform data distribution, optimal data movement when the nodes are added and removed and there is also coexistence of scalability and efficiency in theory. In general the Ideal algorithms use pseudorandom functions and they achieve pseudo-equal data distribution.

This section introduces the operations of ASURA. It takes an input (a datum ID), the maximum segment number plus 1 and the length of each segment. And it outputs the segment number assigned to a data-storing node. ASURA is explained in terms of:

- Basic ASURA:

ASURA’s algorithm can be divided into two steps.
STEP 1. Nodes are assigned to segments in a number line.
STEP 2. A data-storing node is determined.

- ASURA random numbers: ASURA can expand or shrink the range of the random numbers using a special method. ASURA achieves both the qualities scalability and efficiency by using the ASURA random numbers. The random number sequence for ASURA has the following characteristics. Thus, ASURA random numbers also need to have same characteristics.
    1. If the seed (datum ID) is the same, the same random number sequence is generated.
    2. If the seed (datum ID) is not the same, a superficially different random number sequence is generated.
    3. Random numbers in the random number sequence are nearly homogeneously distributed. [4]

- Generation of ASURA random numbers: The algorithm for generating ASURA random numbers uses several pseudorandom number generators. Each pseudorandom number generator has the following specifications.
    1. It generates a different range of random numbers.
    2. The range of random numbers generated by a pseudorandom number generator having a wider range of random numbers inevitably covers the range of random numbers generated by one having a narrower range of random numbers.
    3. If the seed is the same, the same random number sequence is generated.
    4. If the seed is not the same, a superficially different random number sequence is generated.
    5. Random numbers in the random number sequence are nearly homogeneously distributed.

- Node addition and node removal: Firstly, in ASURA, correlating all the nodes with segments is generally centralized, but ASURA can avoid a single point of failure (SPOF).Secondly, for easier calculation of the data-storing node when nodes are added or removed in ASURA, the datum shall have one number (ADDITION NUMBER) in metadata which will accelerate the termination of data that moves to the added node or to recalculate the ADDITION NUMBER when nodes are added and the datum can have N numbers (REMOVE NUMBERS) in metadata to accelerate the determination of moving data when node are removed.

IV. QUANTITATIVE EVALUATION

This section discusses a quantitative evaluation of Hybrid Redundant Data Distribution Algorithm, Load Rebalance Algorithm and ASURA.
It is assumed that capacity of the nodes is fixed because flexible capacity would invite introduce too much complexity for comparison.
Quantitative evaluation poses a considerable problem, i.e. the choice of an algorithm for either hashing numbers or generating random numbers. The results achieved in quantitative evaluation will depend on this choice.

	Calculation Time		Memory Consumption	Uniformity of Distribution	Flexible Data Distribution
	Initial Stage	Distribution			
Hybrid Redundant Data Distribution	**Poor** O(NV x (log(NV)))	**Medium** O(log(NV))	**Poor** O(NV)	**Poor** Double variability	**Medium** Coarsely
Load Rebalance Algorithm	**Good** -	**Poor** O(N)	**Good** O(N)	**Good** Single variability	**Medium** In limited case
ASURA	**Good** -	**Good** O(1)	**Good** O(N)	**Good** Single variability	**Good** Flexibly

Table 1*: Quantitative evaluation of storage methods*

V. CONCLUSIONS

This paper presented a review of various algorithms for distributed cloud storage systems, and we looked at ASURA in particular which was found to be able to distribute data in accordance with node capacity and retain uniformity with minimum data movements when nodes were added and removed. It needs limited resources for its execution and it has enormous scalability. This paper contrasted it with similar algorithms, i.e., Hybrid Redundant Data Distribution, Load Rebalance Algorithm and Data Replication Algorithm. Qualitative and quantitative evaluations were performed to demonstrate their characteristics. Overall, it was found that ASURA is the best distribution algorithm for distributed cloud storage systems.

ACKNOWLEDGEMENT

Our heartfelt thank goes to Mukesh Patel School of Technology Management and Engineering, NMIMS University for providing us with a strong platform to develop our skills and capabilities as well as helping us display our potential as surveyors. I would also like to thank the Head of the Department, my guide and respective teachers for their valuable support and motivation.

REFERENCES

[1] On Cloud Storage and the Cloud of Clouds Approach, Daniel Slamanig, Christian Hanser, The 7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)

[2] Cloud Storage as the Infrastructure of Cloud Computing, Jiyi Wu, Lingdi Ping, Xiaoping Ge,Ya Wang, Jianqing Fu, 2010 International Conference on Intelligent Computing and Cognitive Informatics

[3] A Load Balancing Algorithm For Private Cloud Storage, Prabavathy.B, Priya.K, Chitra Babu, 4th ICCCNT 2013 July 4-6, 2013, Tiruchengode, India

[4] ASURA: Scalable and Uniform Data Distribution Algorithm for Storage Clusters, Ken-ichiro Ishikawa, Cloud System Laboratory, NEC

[5] M. Mense and C. Scheideler, “SPREAD: an adaptive scheme for redundant and fair storage in dynamic heterogeneous storage systems,” In Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms, Philadelphia, USA, January 2008, 1135-1144.

[6] Scalable and Cost-Efficient Algorithms for Reliable and Distributed Cloud Storage, Makhlouf Hadji, Springer International Publishing Switzerland 2016 M. Helfert et al. (Eds.): CLOSER 2015, CCIS 581, pp. 15–37, 2016. DOI: 10.1007/978-3-319-29582-4 2

[7] Thanasis, P.G., Bonvin, N., Aberer, K.: Scalia: an adaptive scheme for efficient multi-cloud storage. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, Los Alamitos, CA, USA, pp. 20: 1–20: 10 (2012)

[8] K. D. Bowers, A. Juels, and A. Oprea, “HAIL: A High-Availability and Integrity Layer for Cloud Storage,” in 16th ACM Conference on Computer and Communications Security, CCS ’09. ACM, 2009, pp. 187–198.

[9] J. Spillner, G. Bombach, S. Matthischke, J. Muller, R. Tzschichholz, and A. Schill, “Information Dispersion over Redundant Arrays of Optimal Cloud Storage for Desktop Users,” in UCC, 2011, pp. 1–8.

[10] J. Myint, T. Naing. ”A data placement algorithm with binary weighted tree on PC cluster-based cloud storage system”. IEEE International Conference on Cloud and Service Computing (CSC), 2011, pp. 315–320.

[11] Improving Storage Availability in Cloud-of-Clouds with Hybrid Redundant Data Distribution, Bo Mao, Suzhen Wu, Hong Jiang, 2015 IEEE 29th International Parallel and Distributed Processing Symposium.