JES
Journal of Engineering Sciences

Vol 13, Issue 02, FEB /2022
ISSN NO:0377-9254

Secure Cloud Storing and Sharing of Big Data Using Data Chunks

[1]*Poojari Anusha*,[2] *D. K. Shareef*

[1]*PG Scholar Department of CSE, P.V.K.K Institute of Technology-Anantapuramu*
[2]*Assistant Professor, Department of CSE, P.V.K.K Institute of Technology- Anantapuramu*

ABSTRACT: *Cloud storage is becoming booming field in the software technology. Cloud storage had made possible to store large file for each enterprises and individual users. However, ancient file systems with intensive optimizations for native disk-based storage backend can't absolutely exploit the inherent options of the cloud to get fascinating performance. At present dramatically increase in the business and internet applications, storage is becoming a major issue in cloud computing. Storage costs are increasing day- by-day. Cloud backed is backing up data that involve distribution copy of the data over a community network to an off-site server. Uncomplicated access interfaces and elastic billing models, cloud storage has become a gorgeous solution to make simpler the storage organization for both enterprises and individual users. This paper presents a survey on the different cloud backed frugal file system. This enables effective storage management, increase the performance and reduce the cost in the cloud. A cloud-back storage system talented of storing and sharing big data in a protected, dependable, and capable. Evaluate Data chunks using apparatus three distinguishing features Cloud storage is becoming booming field in the software technology. Cloud storage had made possible to store large file for each enterprises and individual users. However, ancient file systems with intensive optimizations for native disk-based storage backend can't absolutely exploit the inherent options of the cloud to get fascinating performance. This paper presents a survey on the different cloud backed frugal file system. This enables effective storage management, increase the performance and reduce the cost in the cloud. The cloud is used efficiently in this system and performance is improved.*

Keywords: —*Big-data storage, Cloud storage, Byzantine fault tolerance.*

1. INTRODUCTION

Reinforcement record, information chronicled and cooperation are the mainstream benefits in cloud organizations [1], as a rule these administrations in view of cloud stockpiles like the Amazon S3, Drop box, Google Drive and Microsoft Sky Drive. These administrations are stylish on account of their all over the place availability, pay-as-you-go demonstrate, high capacity, and usability. Such administrations can be for the most part assembled in two modules: (1) individual record synchronization administrations (e.g., Drop Box) Personal document synchronization depends on back-end stockpiling cloud show and the applications of customer speak with the nearby document framework by checking interface [inotify - in Linux]. (2) cloud-sponsored document frameworks (e.g., S3FS [6]). Cloud-sponsored record framework in light of two design models: the First model is intermediary based, second model is open-source arrangements [S3FS [2] and S3QL [3]]. The two models are actualized at client – level. Intermediary based show the intermediary segment put in system foundation, executing as a document server to different customers. Usefulness of Core documents framework is executed as a substitute, to calls the cloud what's more, stores the records. The significant confinement is bottleneck and single purpose of disappointment. Open source

arrangement show the customers straightforwardly get to the cloud, elite of intermediary association as a result, there is no longer a solitary purpose of disappointment, however it's extremely harder to control the document sharing between the customers when miss the reasonable meet point for synchronization. These services are fashionable because of their everywhere accessibility, pay-as-you-go model, high capability, and ease of use. Such services can be generally grouped in two modules: (1) personal file synchronization services (e.g., Drop Box) - Personal file synchronization is based on back-end storage cloud model and the applications of client communicate with the local file system by monitoring interface [inotify -in Linux]. (2) cloud-backed file systems (e.g., S3FS [6]). Cloud-backed file system based on two architecture models: the First model is proxy based, second model is open-source solutions [S3FS [2] and S3QL [3]]. The two models are implemented at user – level. Proxy based model the proxy component placed in network infrastructure, performing as a file server to various clients.

Functionality of Core files system is implemented by proxy, to calls the cloud and stores the files. The major limitation is bottleneck and single point of failure. Open source solution model the clients directly access the cloud, exclusive of proxy interaction as a result, there is no longer a single point of failure, but it's very harder to control the file sharing between the clients when miss the suitable rendezvous point for synchronization. Cloud backup [4] also identified by online backup, is an approach for backing up data that involves a replica of the data over a public network to an off-site system.

Cloud Backed is models that provide data backed up remotely, maintained and managed. Users access the data through the network. Users

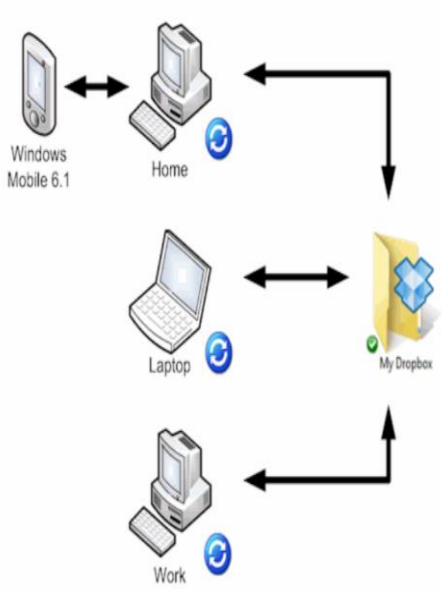

Figure 1: Cloud Backup Services

normally compensate for their data storage on cloud as per-usage or monthly rate. The cloud Storage providers provide a platform as a service, is one of the infrastructure service on cloud storage to shorten storage management for enterprises and personality users. Implementing cloud data backup is able to help boost an organization data protection without raising the workload on information technology.

Online backup systems are classically built a client software application that run on a program determined by the purchase stage of service.

Cloud backups contain the software and hardware component to keep an organization's data, include applications Exchange and SQL Server. Online

backup is used by small and mediumsized businesses (SMBs) and larger enterprises to back up the data. For larger organization, cloud data backup as a complementary form of backup.

II. LITERATURER SURVEY

Distributed File Systems:

CHARON is an intrusion-tolerant file scheme that maintains data privacy, integrity, and availability regardless of the existence of compromised components. Our design adopts some ideas from presented file systems, such as the severance of data and metadata from NASD, volume lease from AFS cloud services instead of communicating directly for coordination.

Data-centric coordination:

A key feature of CHARON is the use of Byzantine-resilient datacentric algorithms for implementing storage space and coordination. There are some works that suggest the use of this kind of algorithms for apply dependable systems [15], [26], [39]. Byzantine disk Paxos [26] is a consensus protocol built on top of untrusted shared disks. More freshly, an enhanced version of this protocol specifically designed to use file synchronization services (e.g., DropBox, Google Drive) instead of disks was published [21]. To the best of our knowledge, there are only two fault-tolerant data-centric lease algorithms in the literature [15] compare with CHARON's BFT composite lease.

Multi-cloud storage:

In the last years, many works have been proposing the use of multiple cloud providers to improve the integrity and availability of stored data [13], [14], [15], [16], [20], [22], [23], [24], [70]. A problem in some of them is the fact they only provide object storage (i.e., read/write registers), which hardens their integration with existing applications. Examples of these systems are RACS [13] for writeonce/archival storage, and DepSky [15], its evolution [16], and ICStore [14] for updatable registers. The main limitation of these systems is that they require servers deployed in the cloud providers, which implies additional costs and management complexity.

FRUGAL CLOUD FILE SYSTEM:

Different embodiment grant a techniques and tools of providing a frugal cloud file system[5] that proficiently uses the blocks of different types of storage devices with special properties for various purposes. The various types of storage strategy reduce the storage and bandwidth transparency. Favorably, the storage and bandwidth reduction in the clouds achieved by the frugal cloud file system, reduce the cost-effective of managing the file system at the same time, maintain high performance. Frugal file system is a structure that optimizes on the whole storage cost between various Cloud storage services and different type of price. The Frugal file system's storage services like a twin storage system, one is

low latency (e.g., Amazon ElastiCache) and the other one is high latency (e.g., Amazon S3). In low latency the data transfer cost is low and cost of storage per byte is high (i.e. cache) but the high latency, cost of storage per byte is low and data transfer cost is high (i.e. disk) Distributed file system (DFS) is the file systems for cloud. DFS allows the clients to access data. Data files are separated by parts as chunks that stored on various remote systems which offer the parallel execution. Data are stored in files in the format of hierarchical tree structure. Directories are denoted by nodes. DFS facilitate any type of enterprises (such as large, medium, small) that allows storing the data and accessing the data on remotely. DFS allow two type of file system as GFS and HDFS. Both file systems are handled the batch processing. Hadoop distributed file system (HDFS) designed for access the terabyte's data or peta bytes data. HDFS is master slave architecture. It consists of Name node and Data node mechanisms. Name node manages the storage of Metadata and Data Node manages the node storage. HDFS file systems the files are divided into blocks. Each block contains various data nodes and every node is replicated for availability. This is the block level replication. Name node manages the operation of name space and map the block to data node. HDFS is characterized by method of In spite of their increasing Technology, current cloudbacked storage systems still have various restrictions related to reliability, durability assurances and inefficient file sharing. SCFS, a cloud-backed file system overcome these challenges and provides strong consistency and POSIX semantics for cloud backed services. It uses a plug gable back plane allows the various cloud storage or a cloud-of-clouds. The main goal of SCFS is assurance of security like integrity, confidentiality, availability and also supports consistency-on-close semantics [28], SCFS is not proposed to be a big-data file system, because file data is downloaded from and uploaded to one or more clouds. SCFS contain the backend cloud storage, co-ordination service and SCFS agent. File data are maintained by backend cloud storage. Metadata management and synchronization supported by co-ordination service. SCFS functionality and client file system mounted by SCFS agent. SCFS provide the strong cloud consistency based on two approaches as maintaining the Meta with limited capacity data and save the data itself. SCFS support two prototype co ordination services: zookeeper [29] and Depspace [13]. The two services are integrated with SCFS wrappers. The co –ordination services simulated the fault tolerance. Zookeeper need 2f + 1 replica for tolerate f crashes by using Paxos-like protocol. Depspace need 3f + 1 or 2f + 1 replica for tolerate f arbitrary faults by using BFT-SMaRt replication engine. SCFS cloud storage services are Amazon S3, Windows Azure Blob, Google Cloud Storage, Racks pace cloud files and all services using cloud of cloud back end. Cloud back end use the extended version of DepSky, support a new operation as read the all versions of hashes data are stored in Depsky's Metadata internal objects and stored in cloud. Based on the consistency and sharing requirements of stored data the SCFS operation divided into 3 modes. 1) Blocking mode 2) Non –blocking mode 3) Non-sharing mode. The main uses of SCFS are backup,

disaster recovery, and file sharing control and without require dependence on any single cloud provider.

DEPSKY: Dependable and Secure Storage in a Cloudof-Clouds DEPSKY, cloud backup improve the availability, integrity and confidentiality of information store in the cloud with help of encoding, replication and encryption of the data on varied clouds that make a cloud-of-clouds. DEPSKY is a reliable and protected storage system that gives the profit of cloud computing by using an arrangement of diverse commercial clouds to cloud-of-clouds. DEPSKY also give the virtual storage, it is accessed by users while invoking the operations. DEPSKY also provide the four limitations such as Loss and corruption of data, Loss of privacy, Vendor lock-in, Loss of availability. DEPSKY System use data and system models. It contains two main algorithms as DEPSKY – A and DEPSKY – CA and also contains the set of auxiliary protocols. Two algorithms are implemented by software library in the clients. Data model contain the three abstraction levels. First level, the conceptual data unit has unique name, version number – support the object updates, data verification – a cryptographic data hash. Second level, Conceptual unit is implemented by generic data unit, has two types of files: signed Metadata file and storage file. Third level, data unit are implemented. Data unit support the operation of storage objects like creation of Metadata file, destruction of data unit, write operation and read operation. System Model uses the asynchronous distributed system; it's composed by writers, readers and cloud storage providers. Quorum protocols can provide as the backbone of storage systems. Quorum protocols contain the individual storage nodes instead of servers. Many protocols involve several steps to access the shared memory, it makes unrealistic for geographically isolated distributed systems such as DEPSKY. The DEPSKY protocols need two communication round-trips to read or write the metadata and data files. Byzantine fault-tolerant (BFT) storage is implemented by several protocols. But its require server for execute code and functions; it's not available on cloud storage. This the key difference between DEPSKY protocols and BFT protocols. DEPSKY – A is the protocol of DEPSKY, it improves the availability and integrity of storage cloud by replication using quorum techniques. DEPSKY -A include read and write algorithm The DEPSKY-A protocol has two major restrictions. First, one is data unit size and costs, the data stored in single cloud. Second one is data storage is clear text, so it does not provide confidentiality guarantee. DEPSKY – CA protocol overcome these problems and also has the additional cryptographic function and coding functions. DepSky-CA write algorithm's encryption technique generates the key sharing.

III. PROPOSED SYSTEM

The proposed cloud backed file system that able to store and share the large amount of data between various cloud providers and cloud storage system in secure, reliable manner. The two main feature of Data chunks is server less design and efficient management of file system. Data Chunks support

three types of data locations as cloud of clouds, public cloud storage and private cloud storage. Cloud of clouds provides mufti cloud availability, confidentiality. Single storage cloud is low cost compared to cloud of clouds but it requires confidence provider. Private cloud storage based on adopted method and solution, also provides the dependability level. Data Chunks concept is data are separated by file data and Metadata. Metadata are stored in cloud of clouds. Data Chunks uses data centric. Byzantine-resilient leasing algorithm which ignores the concurrency conflicts. Data Chunks divides the files into constant size blocks. Files are stored in various data location based on the requirements. POSIX interface is provided by Data Chunks allow the user interact with any file system. Data Chunks cloud storage providers are Amazon S3, Windows Azure Storage, Backspace Cloud Files, and Google Cloud Storage. Data Chunks consists of two design concepts: first design is writes on absorbs file and the second design is remove write – write conflicts and mechanism of ruling out optimistic. Data Chunkis design implementation has main three challenges a: 1) Ability to deal multiple cloud storage locations, 2) Proper file system management and 3) concurrent access to the file system. Data Chunks use modular based approach for non fault tolerant, that build service of cloud.

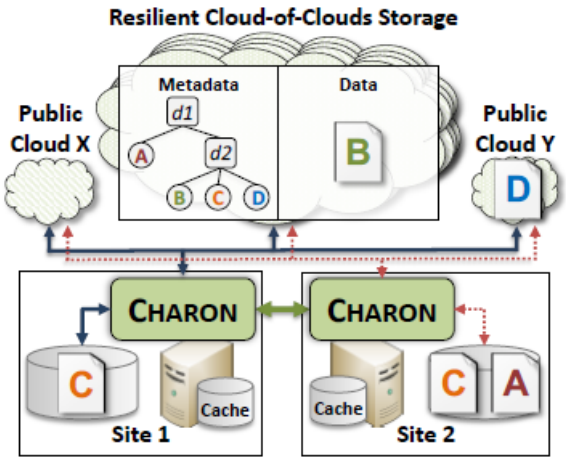

Fig 2. System Architecture

Each client has a unique id, an account for each cloud, and limited local storage. Every cloud provider offers one or more services, which implement access control mechanisms to ensure that only authorized accounts can access them. Data Chunks implements a security model where the owner of the file pays for its storage and defines its permissions. Data Chunks is a distributed file system that provides a near-POSIX interface to access an ecosystem of multiple cloud services and allows data transfer between clients. In particular, the system needs to (1) efficiently deal with multiple storage locations, (2) support reasonably big files, and (3) offer controlled file sharing. Data Chunks separates file data and metadata in different objects stored in diverse locations and manages them using different strategies, as illustrated in Figure 2 In a very high level, Data Chunks interacts with the clouds for three main reasons: (1) storing/retrieving files' data, (2) storing/retrieving file system's metadata, and (3) obtaining/releasing leases to avoid write-write conflicts. Data Chunks is designed around three

Vol 13, Issue 02, FEB /2022
ISSN NO:0377-9254

distributed computing abstractions built on top of basic cloud services. The integrity of a file is attained by cross-verifying the validity of the stored chunks using SHA-256 cryptographic hashes saved in all clouds. Confidentiality is enforced by encrypting the data with a randomly generated AES 256-bit length key and storing it in a secure cloud-based single-writer multi-reader register. The fundamental differences are: (1) the encryption key is split into shares (using Shamir's secret sharing) that are stored with each encoded block , and (2) read/write operations require two cloud accesses (one for the metadata file, and another for the dataitself).

IV METHODOLOGY

The new advanced encryption standard algorithm must be a building block cipher capable of conduct 128 bit blocks, using keys sized at 128, 192, and 256 bits; other principle for being select as the next advanced encryption standard algorithm included:

Security: challenging algorithms were to be judge on their skill to oppose attack, as compare to other submit ciphers, though security strength was to be considered the most important feature in the competition.

Cost: proposed to be released under a global, nonexclusive and royalty-free basis, the candidate algorithms were to be evaluate on computational and memory effectiveness.

 Implementation: Algorithm and completion characteristics to be evaluated included the agility of the algorithm; suitability of the algorithm to be implemented in hardware or software; and overall, relative plainness of implementation

AES encryption works:

AES comprises three chunk ciphers: AES-128, AES-192 and AES-256. Each cipher encrypts and decrypts data in block of 128 bits using cryptographic keys of 128-, 192- and 256-bits, respectively. The Rijndael cipher was intended to believe additional block sizes and key lengths, but for AES, those functions were not adopted

Figure:3 AES ALGORITHM

SHA algorithm:

Secure Hash Algorithms, also known as SHA, are a relations of cryptographic functions designed to keep data protected. It works by transform the data

with a hash function: an algorithm that consists of bitwise operations, modular additions, and compression functions. The hash function then produces a fixed-size string that looks nil like the original. These algorithms are designed to be one-way functions, meaning that once they're transformed into their individual hash values, it's virtually unfeasible to transform them back into the unique data. A few algorithms of interest are SHA-1, SHA-2, and SHA-3, each of which was successively designed with increasingly stronger encryption in reply to hacker attacks. SHA-0, for order, is now superseded due to the broadly bare vulnerabilities. Data Chunks is a user-space file system implemented using FUSE-J, a Java wrapper for the FUSE library. The system is fully implemented at the client side, using cloud services for storage and coordination, and is publicly available as open-source software.

Metadata Organization:

Metadata is the set of attribute assigned to a file/directory. Independently of the position of the data chunks, Data Chunks stores all metadata in the cloud-of-clouds using single-writer multi-reader registers to improve their accessibility and ease of use guarantees. More specifically, we redesigned and optimized the SWMR register execution of DepSky [15] to improve the presentation and concurrency

Managing namespaces:

All metadata is store within namespace objects, which encapsulate the hierarchical arrangement of files and directories in a subdirectory tree. Data Chunks use two types of namespaces: personal namespace (PNS) and shared namespace (SNS). A PNS stores the metadata for all non-collective objects of a client. client has access to as many SNSs as the collective folders it can access. Each collective folder is associated with exactly one SNS, which is referenced in the PNSs of the clients allocation it. 4.1.2 Dealing with shared files: The PNS's metadata is downloaded from the cloud-of-clouds only once when the file system is mount. SNSs, on the other hand, need to be periodically fetch to obtain metadata update on collective directories. Client Y can concurrently execute read-only operations on the lease SNS while the client X is writing.

Data Management

The most vital techniques Data uses to administer big files capably.

Multi-level cache:

This system uses the local disk to cache the most fresh files used by clients. it also keeps a fixed small main-memory cache to recover data accesses over open files. Both of these caches execute least freshly used (LRU) policies.

Working with data chunks:

Vol 13, Issue 02, FEB /2022
ISSN NO:0377-9254

Managing big files in cloud-backed file systems bring two main challenges. First, reading (resp. writing) whole (big) files from the cloud is impractical unpaid to the lofty downloading (resp. uploading) latency [24]. Second, big files might not fit in the (memory) cache effective in cloud-backed file system for ensuring working presentation [23], [24]. addresses these challenge by splitting (large) files into fixed-size chunks of 16MB, which results in blocks with a few megabytes after solidity and erasure codes. This small size has been reported as having a good tradeoff between latency and throughput [15], [24] .figure 4.

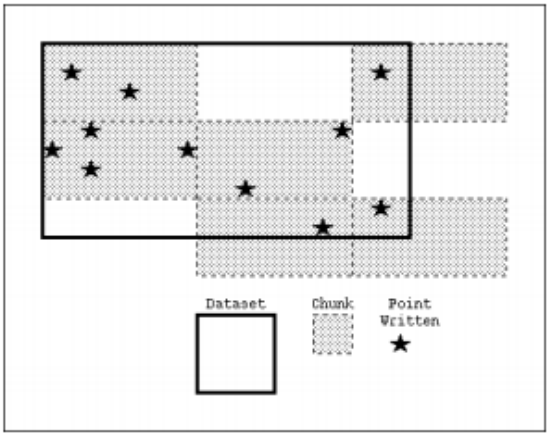

Fig 4 : Proposed Data Chunks

Paper implements a sanctuary model where the owner of the file pays for its storage space and is able to define its permissions. This means that each client pays for all covert data and all the shared data associated with the shared folders he bent (independently on who wrote it). Paper clients are not requisite to be trust since access control is perform by the cloud providers, which implement the permissions for each object. Moreover, the cloud-of clouds admission control is satisfied even if up to f cloud provider misbehave. This happens because if an object is read from up to f faulty providers, no useful in sequence will be obtained (recall that data is encrypted and keys are store using sneaky sharing in a SWMR register). The performance of this model want a mapping between the file system and cloud storage space abstractions.

Service	#A	Costs (μ\$)	Progress
AWS S3	3	15	Obstruction-Freedom
Google Storage	3	15	Obstruction-Freedom
Azure Blob Storage	3	15.6	Obstruction-Freedom
RackSpace Files	3	0	Obstruction-Freedom
Azure Queue	3	1.2	Deadlock-Freedom
RackSpace Queue	3	3	Deadlock-Freedom
AWS DynamoDB	2	subs.*	Deadlock-Freedom
Google Datastore	4	2.4	Obstruction-Freedom

Table showing Cost of Various Services in Cloud

Operation	ext4	NFS	S3QL	SCFS	CHARON
Create	2618	192	105	2	485
Delete	1895	2518	486	4	1258
Stat	15299	20881	5995	9	12925
MakeDir	14998	16664	4242	14	13665
DeleteDir	11998	6785	950	5	8665
ListDir	18759	17426	604	6	9894

Table Showing Meta Data Comparison with various algorithms

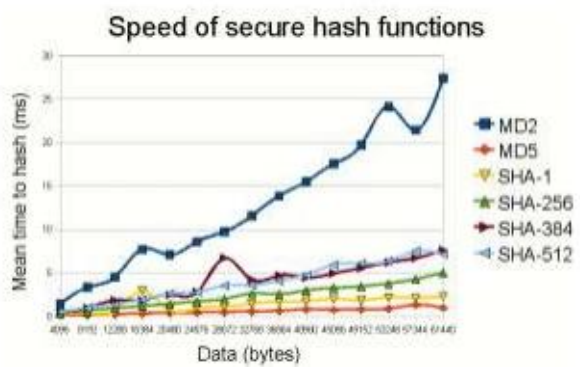

Fig 5: Speed Secure hash Comparison

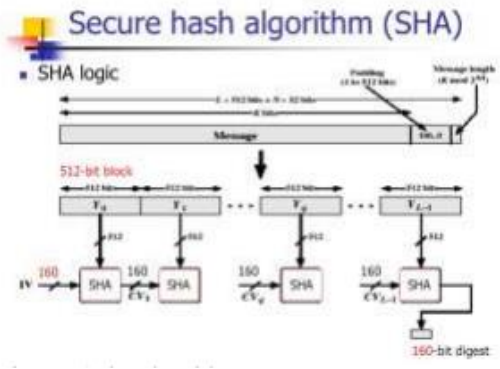

Fig 6: Secure hash Algorithm

Figure.6 suggests the accuracy of question end result received with the aid of query-issuing node. The X-axis denotes the quantity of requested information gadgets and Y-axis denotes the accuracy. The proposed pinnacle-k question technique will increase the accuracy even when the range of asked records gadgets is big. Figure.6 shows the traffic took place when question results are forwarded in more than one routes. The X-axis denotes the wide variety of requested data items and Y-axis denotes the visitors. Figure.Three shows the malicious node identity ratio that represents maximum wide variety of identified malicious node with the aid of issuing less range of queries. The X-axis denotes the query issuing time and misidentification. The suggest approach suggests the question end result improves when the malicious nodes are recognized and removed and additionally while the malicious nodes are gift the query end result accuracy is low as shown in determine 2. The parent 3 indicates the site visitors float whilst the queries are issued inside the community and it's far in comparison with the attack and without assault. The site visitors is excessive while there may be a malicious nodes inside the community on account that they make a contribution false facts inside the community this lead the regular node to send extra query to relax on correct result.

V. CONCLUSION

This paper presented a survey on the different frugal file system for cloud backed services. The frugal cloud based file system improves the performance and cost for end users. Frugal cloud back up is the combination of intellectual data backup & recovery and simple unified solution that safe the organization data. It provides the organization's management services, disaster recovery plan, energy efficiency and cost reduction. Chunks are cloud-backed file system for storing and sharing big data. Its design relies on two important principles: files metadata and data are store in numerous clouds, without require trust on any of them independently, and the system is entirely datacentric. This design has led us to

expand a novel Byzantineresilient leasing protocol to avoid write-write conflicts without any custom server. Our results show that this design is viable and can be employed in real-world institutions that need to store and share big critical datasets in a controlled way.

REFERENCE

[1] Future of cloud computing - 2nd annual survey results.http://goo.gl/fyrZFD, 2012.

[2] S3FS - FUSE-based file system backed by Amazon S3.

[3] http://code.google.com/p/s3fs/.

[4] S3QL - a full-featured file system for online data storage.

[5] http://code.google.com/p/s3ql/.

[6] [http://searchcloudstorage.techtarget.com/definition

[7] Krishna P.N. Puttaswamy, Thyaga Nandagopal and Murli kodialam "Frugal storage for cloud file system," in proceeding EuroSys'12 of the 7th ACM European conference on computer Systems,2015 pages 71-84.

[8] https://en.wikipedia.org/wiki/Distributed

[9] Michael Vrable, Stefan Savage, and Geoffrey M. Voelker "Blue Sky: A Cloud-Backed File System for the Enterprise, "in Proceedings of the 10th USENIX conference on File and Storage Technologies, Feb 2012.

[10] M. Rosenblum and J. K. Ousterhout."The Design and Implementation of a Log-Structured File System," ACM Transactions on Computer Systems (TOCS), 1992.

[11] Alysson Bessani, Ricardo Mendes, Tiago Oliveira, Nuno Neves, Miguel Correia, Marcelo Pasin, and Paulo Verissimo "SCFS: A Shared Cloud-backed File System, " in Proceedings of the USENIX ATC on USENIX Annual Technical Conference 19&20-June -2014.

[12] P. Mahajan, S. Setty, S. Lee, A. Clement, L. Alvisi, M. Dahlin, and M. Walfish "Depot: Cloud Storage with Minimal Trust," In Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI), Oct. 2010.

[13] J. Howard "Scale and performance in a distributed file system" ACM Trans. Computer Systems, 1988.

[14] P. Hunt, M. Konar, F. Junqueira, and B. Reed. "Zookeeper: Waitfree coordination for internet-scale services," In USENIX ATC, 2010.

[15] A. Bessani, E. P. Alchieri, M. Correia, and J. S. Fraga "DepSpace:A Byzantine fault-tolerant coordination service," in EuroSys, 2008.

[16] StorSimple. StorSimple. http://www.storsimple.com/.

[17] TwinStrata. TwinStrata. http://www.twinstrata.com/.

[18] Alysson Bessani, Miguel Correia, Bruno Quaresma, Fernando Andr´e, and Paulo Sousa "DEPSKY: Dependable and Secure Storage in a Cloud-of-clouds," EuroSys 11-April-2011.

[19] Ricardo Mendes, Tiago Oliveira, Vinicius Cogo, Alysson Bessani " The CHARON file system,"

[20] Idilio Drago, Marco Mellia, Maurizio M. Munafò, Anna Sperotto and Aiko Pras "Inside Drop box: Understanding Personal Cloud Storage Services," in Proceeding of IMC -12 of ACM conference on internet measurement conference,2012 PP.481-494.

[21] http://www.techrepublic.com/blog/five-apps/keep-your-data-safewith-one-of-these-five-cloud-backup-tools/ [22] http://www.cloudwards.net/spideroak-or-wuala-which-is-moresecure/

[23] Kailas Pophale, Priyanka Patil, Rahul Shelake, Swapnil Sapkal "Seed Block Algorithm: Remote Smart Data- Backup Technique for Cloud Computing," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 3, March 2015.

[24] Lili Sun, Jianwei An, Yang, and Ming Zeng "Recovery Strategies for Service Composition in Dynamic Network," International Conference on Cloud and Service Computing,2011

[25] Giuseppe Pirr´o, Paolo Trunfio, Domenico Talia, Paolo Missier and Carole Goble "ERGOT: A Semantic-based

System for Service Discovery in Distributed Infrastructures," 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010.

[26] Xi Zhou, Junshuai Shi, YingxiaoXu, Yinsheng Li and Weiwei Sun "A backup restoration algorithm of service composition in MANETs," Communication Technology ICCT 11th IEEE International Conference, 2008, pp. 588-591.

[27] Ms...KrutiSharma, and Prof K.R. Singh "Online data Backup and Disaster Recovery techniques in cloud computing: A review", JEIT, Vol.2, Issue 5, 2012.

[28] Eleni Palkopoulouy, Dominic A. Schupke, Thomas Bauscherty "Recovery Time Analysis for the Shared Backup Router Resources (SBRR) Architecture", IEEE ICC, 2011.

[29]http://searchcloudstorage.techtarget.com/definition/cloud-storage