SECURE CLOUD STORING AND SHARING OF BIG DATA USING DATA CHUNKS

Koilakuntla Munni[1] & Kodi Ramadevi[2]

[1]*PG Student, Annamacharya Institute of Technology and Sciences, Kadapa, Andhra Pradesh.*

[2]*Asst. Professor, Annamacharya Institute of Technology and Sciences, Kadapa, Andhra Pradesh.*

ABSTRACT

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. CHARON, a cloud-back storage system talented of storing and sharing big data in a protected, dependable, and capable. CHARON apparatus three distinguishing features: (1) it does not require trust on any single article, (2) it does not want any client-managed server, and (3) it efficiently deals with big files over a set of geo-discrete storage services. As well that, we urban a novel Byzantine-resilient data-centric rental protocol to avoid write-write conflict between clients accessing shared repositories. We evaluate CHARON using micro and application-based benchmark simulate representative. The results design is not only viable but also present an end-to-end routine of up to 2:5 better than other cloud-backed solution.

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

INTRODUCTION :-

The high volume, velocity, and variety of of data organization, requiring them to scale while make sure security and data being bent by diverse scientific and business domain challenge standard solution dependability. We here CHARON, a near-POSIX cloud-backed storage space system capable of storing and sharing big data with minimal organization and no devoted infrastructure. The main motivation for building this system was to support the organization of genomic data, the use of widely-accessible cloud services would facilitate the sharing of data among biobanks, hospitals, and laboratories, serving as a managed repository for public and access-controlled datasets. The problem is how to exploit the benefits of public clouds for data storage and sharing without endangering the security and dependability of biobanks' data. CHARON uses cloud-of-clouds replication [13], [14], [15],[16] of encrypted and encoded data to shun having any cloud Ensure. Backup file, data archival and collaboration are the popular services in cloud companies [1], in general these services based on cloud storages like the Amazon S3, Drop box, Google Drive and Microsoft Sky Drive. 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

Figure 1: Cloud Backup Services

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

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.

PROJECT OBJECTIVES:-

Present CHARON, a near-POSIX cloud-backed storage system capable of storing and sharing big data with minimal management and no dedicated infrastructure. The main motivation for building this system was to support the management of genomic data, as required by bioinformatics and life sciences organizations. Furthermore, CHARON is capable of handling big data in a secure way by dividing files into chunks, employing encryption, erasure codes, and compression, and using prefetching and background uploads. The way integrate these techniques into a usable system makes CHARON unique, both in terms of design and offered features. Furthermore, the end-to-end performance of CHARON is 2–4better than competing multi-cloud systems, offering a usage experience as good as standard NFS. In summary, the paper contributions are The design and implementation of CHARON, a practical cloud-backed storage system for storing and sharing big data (x2 and x4); A Byzantine-resilient data-centric lease algorithm that exploits different cloud services without requiring trust on any of them individually (x3); An evaluation comparing CHARON with local, networked, and cloud-backed storage systems, using microbenchmarks and a novel benchmark that captures the I/O of bioinformatics applications (x5).

CHARON separates file data and metadata in different objects stored in diverse locations and manages them using different strategies, as illustrated in Figure 1. File data locations are of three kinds in CHARON: cloud-of-clouds, single (public) storage cloud, and private repository (e.g., a private cloud). These alternatives explore various cost-dependability tradeoffs and address all placement requirements have encountered with life sciences and big data applications.

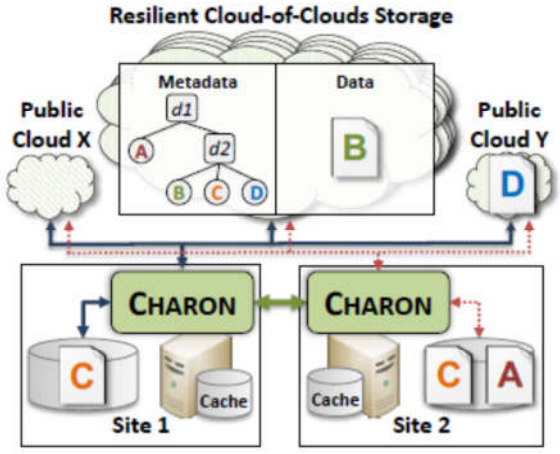

Fig 1 System Architecture

For example, the cloud-of-clouds can store critical data (CHARON'S namespace and file B) that needs the availability and confidentiality assured by the multi-cloud scenario (provider-fault tolerance). A single cloud can store noncritical public studies and anonymized datasets (file D) (providerdependent and potentially less expensive). Finally, private repositories must be used to keep clinical data from human samples that cannot leave the boundaries of a particular institution (file A) or country (file C) (subject to local infrastructure restrictions). CHARON maintains the namespace tree, together with the files' metadata, replicated in the cloud-of-clouds storage. The rationale for this decision is to keep the file system structure secure by exploiting the expected high availability of cloud-of-clouds and by expanding on the efficient data-centric replication protocols developed in the last years. The objective is to have only soft state in clients, which can be reconstructed after a crash by fetching data from the clouds. In a very high level, CHARON 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. The order of these interactions depends on the operation a client is performing. In a write, the client obtains the lease, uploads the files' data, and uploads the corresponding metadata. In a read, the client obtains the file system's metadata and then downloads the data associated with the requested files. The next section describes the computing abstractions these interactions rely on, while Section 4 details how the described operations works.

PROPOSED SYSTEM:-

CHARON is one of the 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 CHARON is server less design and efficient management of file system. CHAON 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. CHARON data are separated by file data and Metadata. Metadata are stored in cloud of clouds. CHARON use data centric. Byzantine-resilient leasing algorithm which ignores the concurrency conflicts. CHARON divides the files into constant size blocks. Files are stored in various data location based on the requirements. POSIX interface is provided by CHARON that allow the user interact with any file system. CHARON cloud storage providers are Amazon S3, Windows Azure Storage, Backspace Cloud Files, and Google Cloud Storage. CHARON 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. CHARON 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. CHARON use modular based approach for non fault tolerant, that build service of cloud.

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. CHARON implements a security model where the owner of the file pays for its storage and defines its permissions. CHARON 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

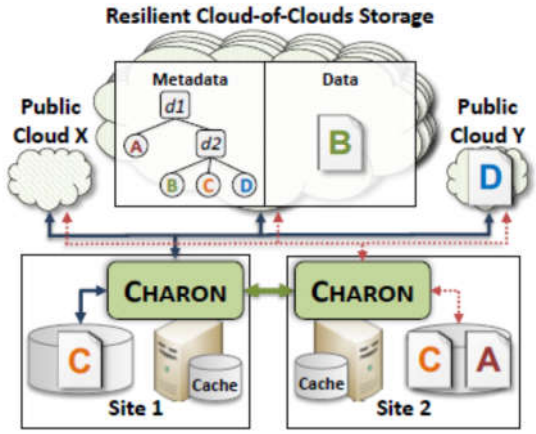

Fig 2. System Architecture

. In particular, the system needs to (1) efficiently deal with multiple storage locations, (2) support reasonably big files, and (3) offer controlled file sharing. CHARON 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, CHARON 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. CHARON is designed around three 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 data itself).

.

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. CHARON 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, CHARON 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. CHARON 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 CHARON uses to administer big files capably.

Multi-level cache:

CHARON 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:

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]. CHARON 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.

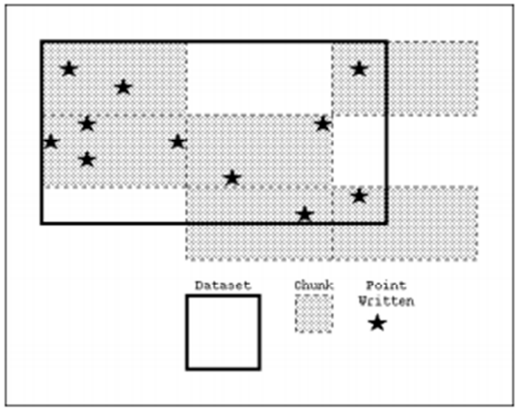

Fig 4 : Charon Data Chunks

CHARON 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). CHARON 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.

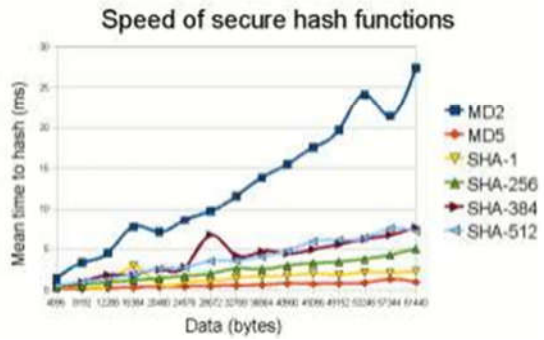

Fig 5: Speed Secure hash Comparison

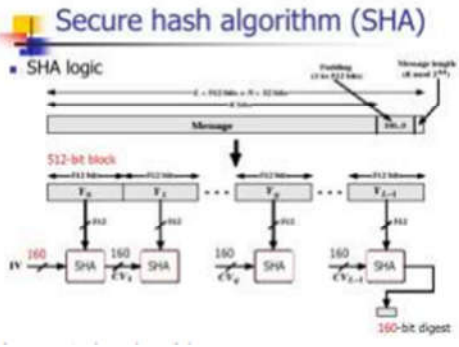

Fig 6: Secure hash Algorithm

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.

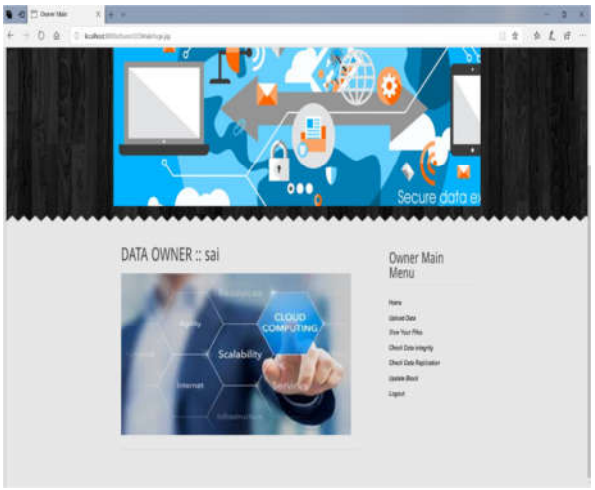

Fig 7: Owner Cloud Activities

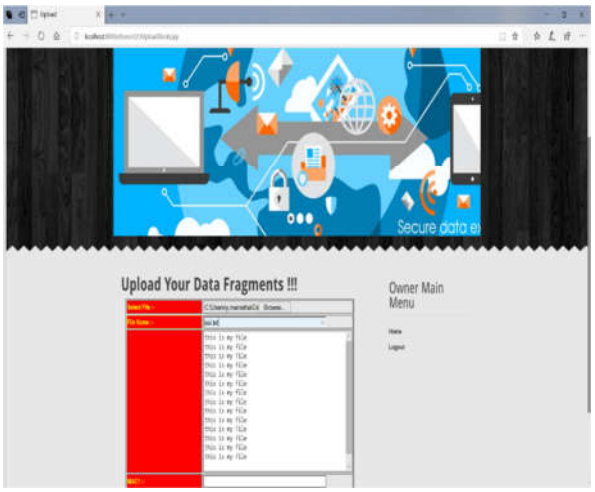

Fig 5: Upload Data

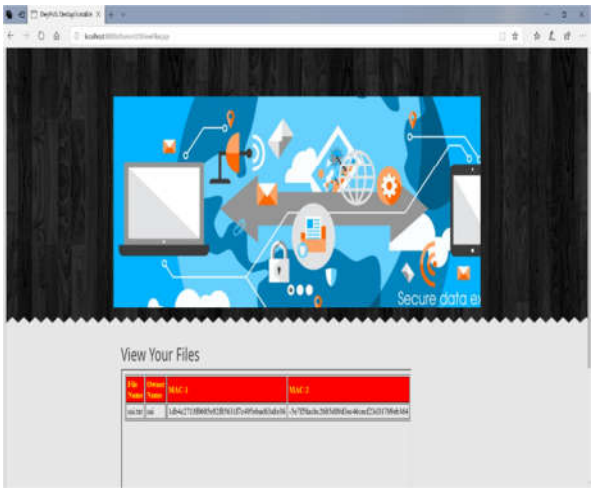

Fig 6: Data in chunks

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. CHARON is a 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