Scientific Journal of Impact Factor (SJIF): 5.71

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 5, Issue 05, May -2018

STORAGE DEDUPLICATION BY USING MAP REDUCE

Ayushi Osatwal[1], Priyanka Bhamre[2], Srushti Patil[3], Swapnali Shirke[4]

Department of Information Technology, JSPM's Rajarshi Shahu College of Engineering, Pune,
Department of Information Technology, JSPM's Rajarshi Shahu College of Engineering, Pune,
Department of Information Technology, JSPM's Rajarshi Shahu College of Engineering Pune,
Department of Information Technology, JSPM's Rajarshi Shahu College of Engineering, Pune,

Abstract - *Information are created and updated rapidly by users through any device at any place. Adapting to this multiform information continuously is an overwhelming test. Hadoop circulated document system (HDFS) is intended to manage data for building an appropriated server farm. HDFS utilizes the information copies to build data unwavering quality. However, data copies require a great deal of additional storage room and financing in system. Utilizing the deduplication procedure can enhance use of the storage room viably. In this paper, we propose a dynamic deduplication choice to enhance the capacity use of a server which utilizes HDFS as its document framework. Our proposed system can define a legitimate Deduplication procedure to adequately use the storage room under the restricted stockpiling gadgets. This deduplication methodology erases pointless copies to build the storage room. The exploratory outcomes demonstrate that our technique can proficiently enhance the capacity use of a server utilizing the HDFS system.*

Keywords- *Cloud storage, data deduplication, Hadoop distributed file system.*

I. INTRODUCTION

In the present condition because of expanding interest of web empowered gadgets we are getting gigantic measure of information. Presently, information develops quickly from numerous social destinations and media utilized by people groups like Facebook twitter satellites, Airplanes, stock advertising and so forth. They all are producing part of information every second and all information put away and associated with cloud. Examination and preparing of such kind of information is very troublesome [1]. It winds up hard to process this vast measure of information utilizing close by Database administration apparatuses or conventional data handling application. To conquer this sort of issue Hadoop is to be utilized. Hadoop gives Hadoop Distributed File System (HDFS) that can deal with unstructured information effectively. In enormous information condition deduplication and its procedures were utilized to expel copy information from huge information [2].

With the touchy development of advanced information, deduplication systems are broadly utilized to reinforce information and limit system and capacity overhead by distinguishing and taking out excess among information. Rather than keeping different information duplicates with a similar substance, deduplication dispenses with repetitive information by keeping just a single physical duplicate and alluding other excess information to that duplicate. Deduplication has got much consideration from both scholarly community and industry since it can incredibly enhances stockpiling use and spare storage room, particularly for the applications with high deduplication proportion, for example, chronicled capacity system.

II. PROPOSED SYSTEM

we propose another system stockpiling framework, "DeDu", to store information that is anything but difficult to share, and in the meantime, to spare storage room, not withstanding for copied reproductions. In DeDu, when a client transfers a document out of the blue, the framework records this record as source information, and the client will get a connection petition for this client himself, and the same for other potential clients, to get to the source information. At the point when the source information has been put away in the framework, if similar information is transferred by different clients, the framework won't acknowledge indistinguishable information from new, yet rather, the new client, who is transferring information, will get a connection document to the first source information. Clients are permitted to peruse the source information however not to compose. Once the underlying client changes the first information, the framework will set the changed information as another one, and another connection document will be given to the client. Alternate clients who interface with the first document won't be affected [24]. Under these conditions, the more clients share similar information, the more storage room will be spared. To erase the duplications, the initial step is to recognize the duplications. The two ordinary approaches to recognize. Duplications are: contrasting information squares or documents a tiny bit at a time and looking at information pieces or records by hash esteems. To think about pieces or records a tiny bit at a time would ensure precision, at the cost of extra time utilization. To analyze squares or records by hash esteem would be more productive, yet the odds of unintentional impacts would be expanded. The shot of coincidental impact relies upon the hash calculation. Be that as it may, the odds are minute. In this manner, utilizing a mix hash an incentive

to distinguish the duplications will incredibly lessen the crash likelihood. Along these lines, it is satisfactory to utilize a hash capacity to recognize duplications [4, 15].

ADVANTAGES OF PROPOSED SYSTEM:

- Higher reliability in which the data chunks are distributed node.
- Security requirements of data confidentiality and tag consistency are also achieved by introducing a deterministic secret sharing scheme in distributed storage systems

III. LITERATURE SERVE

1. Reclaiming Space from Duplicate Files in a Serverel Distributed File System
AUTHORS: John R. Douceur
The Farsite distributed file system provides availability by replicating each file onto multiple desktop computers. Since this replication consumes significant storage space, it is important to reclaim used space where possible. Measurement of over 500 desktop file systems shows that nearly half of all consumed space is occupied by duplicate files. In this present a mechanism to reclaim space from this incidental duplication to make it available for controlled file replication. This mechanism includes 1) convergent encryption, which enables duplicate files to coalesced into the space of a single file, even if the files are encrypted with different users' keys, and 2) SALAD, a Self- Arranging, Lossy, Associative Database for aggregating file content and location information in a decentralized, scalable, fault-tolerant manner. Large-scale simulation experiments show that the duplicate-file coalescing system is scalable, highly effective, and fault-tolerant.

2. DupLESS: Server-Aided Encryption for Deduplicated Storage
AUTHORS: Mihir Bellare

Cloud storage service providers such as Dropbox, Mozy, and others perform deduplication to save space by only storing one copy of each file uploaded. Should clients conventionally encrypt their files, however, savings are lost. Message-locked encryption (the most prominent manifestation of which is convergent encryption) resolves this tension. However it is inherently subject to brute-force attacks that can recover files falling into a known set. This propose an architecture that provides secure deduplicated storage resisting brute-force attacks, and realize it in a system called DupLESS. In DupLESS, clients encrypt under message-based keys obtained from a key-server via an oblivious PRF protocol. It enables clients to store encrypted data with an existing service, have the service perform deduplication on their behalf, and yet achieves strong confidentiality guarantees. We show that encryption for deduplicated storage can achieve performance and space savings close to that of using the storage service with plaintext data.

3. Message-Locked Encryption and Secure Deduplication.
AUTHORS: Mihir Bellare
This paper formalizes a new cryptographic primitive, Message-Locked Encryption (MLE), where the key under which encryption and decryption are performed is itself derived from the message. MLE provides a way to achieve secure deduplication (space-efficient secure outsourced storage), a goal currently targeted by numerous cloud-storage providers. This paper provide definitions both for privacy and for a form of integrity that call tag consistency. Based on this foundation, here make both practical and theoretical contributions. On the practical side, this paper provide ROM security analyses of a natural family of MLE schemes that includes deployed schemes. On the theoretical side the challenge is standard model solutions, and we make connections with deterministic encryption, hash functions secure on correlated inputs and the sample-then-extract paradigm to deliver schemes under different assumptions and for different classes of message sources. This work shows that MLE is a primitive of both practical and theoretical interest.

4. Proofs of Ownership in Remote Storage Systems

Author: Shai Halevi, Danny Harnik

Cloud storage systems are increasingly popular nowadays, and a promising technology to keep their cost down is deduplication, namely removing unnecessary copies of repeating data. Moreover, client-side deduplication attempts to identify deduplication opportunities already at the client and save the bandwidth in uploading another copy of an existing file to the server. In this work identify attacks that exploit client-side deduplication, allowing an attacker to gain access to potentially huge files of other users based on a very small amount of side information. For example, an attacker who knows the hash signature of a file can convince the storage service that it owns that file, hence the server later lets the attacker download the entire file. To overcome such attacks, here introduce proofs-of-ownership (PoWs), where a client proves to the server that it actually holds the data of the file and not just some short information about it. Here formalize proof-of-ownership, present solutions based on Merkle trees and specific encodings, and analyze their security. We

implemented one variant of the scheme, this performance measurements indicate that our protocol incurs only a small overhead (compared to naive client-side deduplication that is vulnerable to the attack).

5. Optimizing Cauchy Reed-Solomon Codes for Fault-Tolerant Storage Applications
Author: James S. Plank

In the past few years, all manner of storage systems, ranging fromdisk array systems to distributed and widearea systems, have started to grapple with the reality of tolerating multiple simultaneous failures of storage nodes. Unlike the single failure case, which is optimally handled with RAID Level-5 parity, the multiple failure case is more difficult because optimal general purpose strategies are not yet known. Erasure Coding is the field of research that deals with these strategies, and this field has blossomed in recent years. Despite this research, the decades-old strategy of Reed-Solomon coding remains the only space-optimal (MDS) code for all but the smallest storage systems. The best performing implementations of Reed-Solomon coding employ a variant called Cauchy Reed-Solomon coding, developed in the mid 1990's [BKK+95]. This paper present an improvement to Cauchy Reed-Solomon coding that is based on optimizing the Cauchy distribution matrix. Here detail an algorithm for generating good matrices and then evaluate the performance of encoding using all manners of Reed Solomon coding, plus the best MDS codes from the literature.
The improvements over the original Cauchy Reed-Solomon codes are as much as 83% in realistic
scenarios, and average roughly 10% over all cases that tested.

IV. MATHEMATICAL MODULE :

Let S be the Whole system which consists,

S= {I,P,O}

Where,

I-Input,

P- procedure,

O- Output.

I-{F,U}

F-Files set of {F1, F2,…., FN}

U- No of Users {U1, U2,……,UN}

Procedure(P):

P={POW, n , , , ϕ, i,j ,m , k}.

Where,

1. POW - proof of ownership.
2. n - No of servers.
3. proof of ownership in blocks.
4. – proof of ownership in files
5. ϕ - tag.
6. i- Fragmentation.
7. j- No of server.

8. m-message

9. k- Key.

A) File Upload(FU):

Step 1: File level deduplication

If a file duplicate is found, the user will run the POW protocol POWF with each S-CSP to prove the file ownership. For the *j*-th server with identity *idj*, the user first computes

$\phi F; idj = \text{TagGen}'(F, idj)$

And runs the PoW proof algorithm with respect to $\phi F, idj$. If the proof is passed, the user will be provided a pointer for the piece of file stored at *j*-th S-CSP. Otherwise, if no duplicate is found, the user will proceed as follows:

First divides *F* into a set of fragments $\{Bi\}$ (where $i = 1, 2, \cdots$).

For each fragment *Bi*, the user will perform a block-level duplicate check.

Step 2: Block Level deduplication

If there is a duplicate in S-CSP, the user runs PoWBon input:

$\phi Bi; j = \text{TagGen}'(Bi, idj)$

With the server to prove that he owns the block *Bi*. If it is passed, the server simply returns a block pointer of *Bi* to the user. The user then keeps the block pointer of *Bi* and does not need to upload *Bi*.

B) Proof of ownership (POW):

Step 1: compute and send ϕ' to the verifier.

Step 2: present proof to the storage server that he owns *F* in an interactive way with respect to ϕ' The PoW is successful if the proof is correct

$\phi' = \phi(F)$

C) File Download (FD)-

To download a file *F*, the user first downloads the secret shares $\{cij,mfj\}$ of the file from *k*out of *n* storage servers. Specifically, the user sends all the pointers for *F* to *k* out of *n* servers. After gathering all the shares, the user reconstructs file *F*, *macF*by using the algorithm of Recover($\{\cdot\}$). Then, he verifies the correctness of these tags to check the integrity of the file stored in S-CSPs.

Output (O):

User can upload, download, recover, share files on cloud server and provide data dedupliaction and reliability.

V. EXISTING SYSTEM

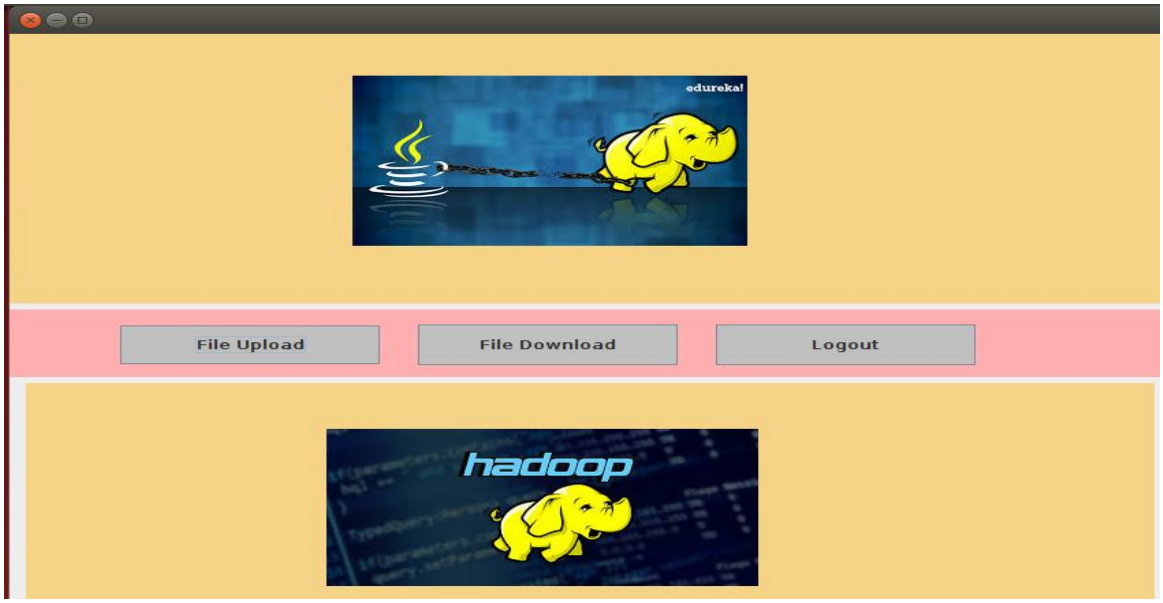
In spite of the way that DE duplication system can save the storage space for the capacity organization providers, it diminishes the resolute nature of the structure. Data steadfast quality is extremely an extraordinarily fundamental issue in a DE duplication amassing structure in light of the fact that there is one and copy for each report set away in the server shared by each one of the proprietors. If such a typical record/protuberance was lost, an exorbitantly significant measure of data gets the opportunity to be hard to reach in perspective of the detachment of the extensive number of reports that offer this archive/piece. If the estimation of a bump were estimated similarly as the measure of record data that would be lost if there ought to be an event of losing a singular piece, at that point the measure of customer data lost when a piece in the limit structure is demolished creates with the amount of the mutual normal for the irregularity. Along these lines, how

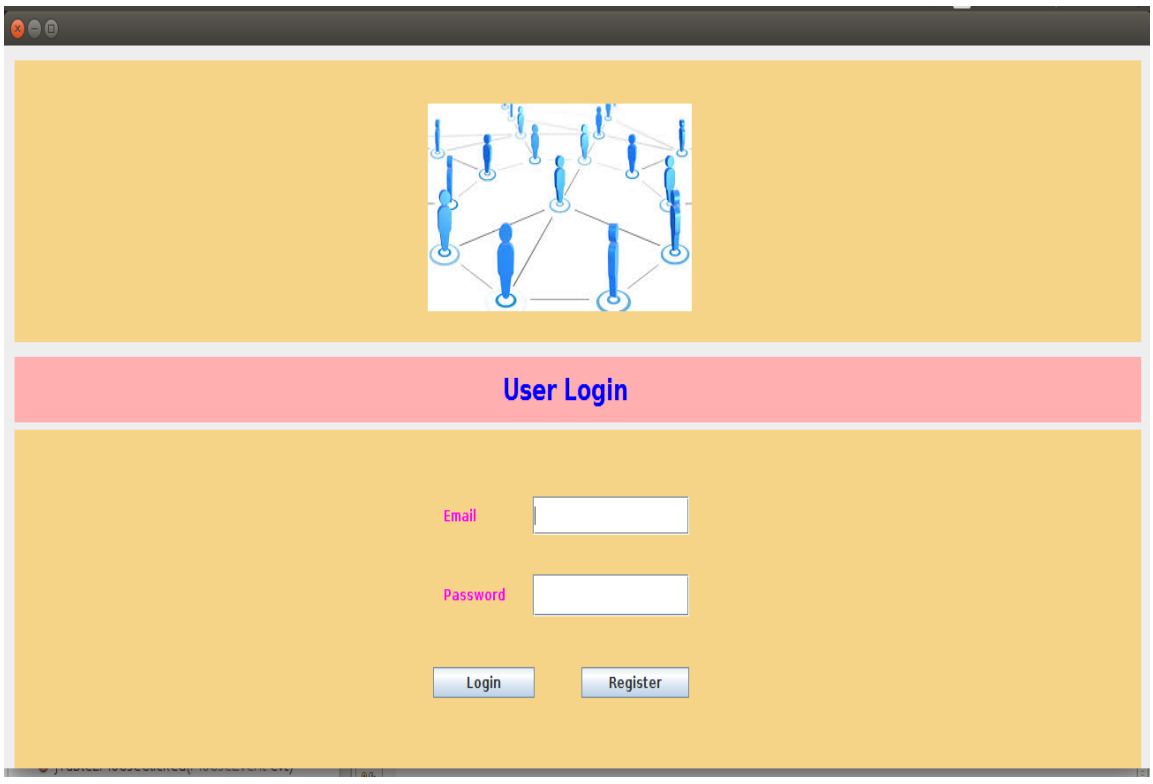
to guarantee high data steadfastness in deduplication structure is a segregating issue. Most by far of the past deduplication structures have quite recently been considered in a single server setting.

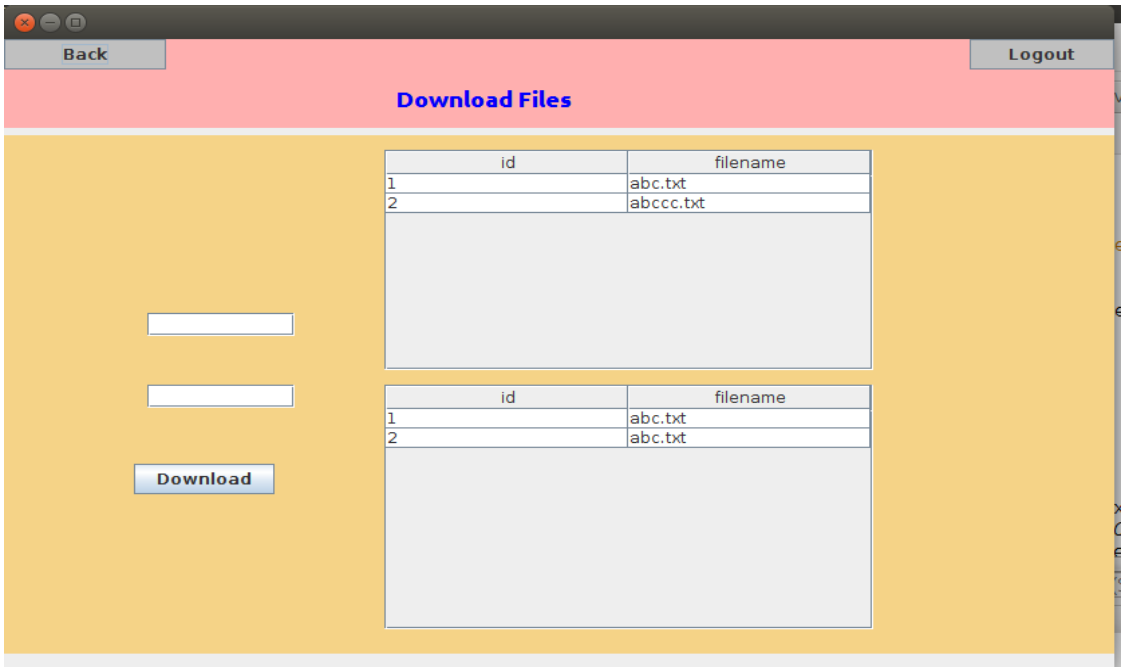
DISADVANTAGES OF EXISTING SYSTEM:

- Block level deduplication cannot be performed.
- System improves storage utilization while reducing reliability.

VI. Result







VII. CONCLUSION

Getting to unstructured information in enormous information is troublesome errand. It is extremely hard to store such a gigantic measure of information and deals with some replication of this information. Enormous information gives deduplication system in which copy information is recognized and expelled on the premise hash estimation of every datum record. There are number of deduplication system, for example, FBC, CDC, Byte record lumping and so forth accessible in huge information. In this paper relative investigation of these procedures has been introduced. After that some hashing methods like MD5, SHA to give hash an incentive to information are likewise examined

REFERENCES

[1] D. Laney, 3D Data Management: Controlling Data Volume, Velocity and Variety. Application Delivery Strategies, META Group, 2001, http://blogs.gartner.com/doug-laney/files/2012/ 01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

[2] S. Ghemawat, H. Gobioff, and S. Leung, "The google file system," in Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03), pp. 29–43, Bolton Landing, NY, USA, October 2003.

[3] Hadoop, http://hadoop.apache.org/.

[4] Sun Microsystems, Take Your Business to a Higher Level: Sun Cloud-Computing Technology Scales Your Infrastructure to Take Advantage of New Business Opportunities, 2009, http://www.aeiseguridad.es/descargas/categoria6/4612546.pdf.

[5] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos et al., "XORing elephants: novel erasure codes for big data," Proceedings of the VLDB Endowment, vol. 6, no. 5, pp. 325–336, 2013.

[6] D. Borthakur, The Hadoop Distributed File System: Architecture and Design, 2007, http://hadoop.apache.org/docs/r0.18.0/ hdfs design.pdf.

[7] Amazon, http://www.amazon.com/.

[8] HBase, http://hbase.apache.org/.

[9] Hive, http://hive.apache.org/..

