

Enhanced Dynamic Whole File De-Duplication (DWFD) for Space Optimization in Private Cloud Storage Backup

M. Shyamala Devi, V. Vimal Khanna, and A. Naveen Bhalaji, *Member, IACSIT*

Abstract—Cloud Storage provide users with abundant storage space and make user friendly for immediate data access. But there is a lack of analysis on optimizing cloud storage for effective data access. With the development of storage and technology, digital data has occupied more and more space. According to statistics, 60% of digital data is redundant, and the data compression can only eliminate intra-file redundancy. In order to solve these problems, De-Duplication has been proposed. Many organizations have set up private cloud storage with their unused resources for resource utilization. Since private cloud storage has limited amount of hardware resources, they need to optimally utilize the space to hold maximum data. In this paper, we discuss the flaws in existing methods for Data De-Duplication. Our proposed method namely Dynamic Whole File De-duplication (DWFD) provides dynamic space optimization in private cloud storage backup as well as increase the throughput and de-duplication efficiency.

Index Terms—Cloud backup, cloud computing, constant-size chunking, data de-duplication, full-file chunking, private storage cloud, redundancy.

I. INTRODUCTION

Cloud computing delivers flexible applications, web services and IT infrastructure as a service over the internet using utility pricing model. The Cloud is a cost-effective approach to technology as there is no need to make usage predictions, upfront capital investments or over purchase hardware or software to meet the demands of peak periods. Cloud computing incorporates virtualization, data and application on-demand deployment, internet delivery of services, and open source software. The different forms of cloud design are Public cloud, Private cloud and Hybrid cloud. **Public clouds** are run by third party service providers and applications from different customers are likely to be mixed together on the cloud's servers, storage systems, and networks. Here the computing infrastructure is hosted by the cloud vendor at the vendor's premises. **Private clouds** are built for the exclusive use of one client. Private clouds can also be built and managed by the organization's own administrator. Here the computing infrastructure is dedicated to a particular organization and not shared with other organizations. Private clouds are more secure when compared to public clouds. Hybrid clouds combine both public and private cloud models for handling planned workload spikes

A. Cloud Storage

Cloud storage is a service model in which data is maintained, managed and backed up remotely and made available to users over a network. Cloud storage [1] provides users with storage space and make user friendly and timely acquire data, which is foundation of all kinds of cloud applications. There are many companies providing free online storage. The storage cloud provides Storage-as-a-Service. The organization providing storage cloud uses online interface to upload or download files from a user's desktop to the servers on the cloud. Typical usage of these sites is to take a backup of files and data. Storage cloud exists for all the types of cloud. A cloud storage SLA is a service-level agreement between a cloud storage service provider and a client that specifies details of the service, usually in quantifiable terms. The forms of cloud storage are private cloud storage, public cloud storage and hybrid cloud storage

B. Advantages of Cloud Storage

Cloud storage has several advantages over traditional data storage. For example, if we store our data on a cloud storage system, we will be able to get that data from any location that has internet access. There is no need to carry around a physical storage device or use the same computer to save and retrieve our information. With the right storage system, we could even allow other people to access the data and turns a personal project into collaborative effort.

C. Private Cloud Storage

Public cloud storage such as Amazon's Simple Storage Service (S3) [2], provide a multi-tenant storage environment that is most suitable for unstructured data. **Private cloud** storage services provide a dedicated environment protected behind an organization's firewall. Private clouds are appropriate for a user who need customization and more control over their data and is shown in Fig. 1. **Hybrid cloud storage** is a combination of at least one private cloud and one public cloud infrastructure. An organization store actively used and structured data in private cloud and unstructured and archival data in a public cloud.

D. Private Cloud Storage Backup

Cloud storage backup [1] is a strategy for backing up data that involves removing data offsite to a managed service provider for protection. A major benefit of using cloud backup is that it can make managing a backup system easier. Data moved offsite should be de-duplicated to avoid the redundancy and it is done by Cloud Storage Controller (CSC). This controller provides data protection, security, advanced virtualization features, and performance for an array of

Manuscript received March 19, 2014; revised May 20, 2014.

The authors are with the R.M.D Engineering College, Chennai, India (e-mail: shyamalapmr@gmail.com, vimalkhanna93@gmail.com, anaveenbhalaji@gmail.com).

locally attached disk drives. The three benefits of CSC are as follows. First, it creates a seamless and highly robust connection to cloud storage, while requiring no changes to applications running in the data center. Applications are able to access the cloud using standard block and file access protocols. Second, it accelerates the performance of applications using cloud storage through advanced WAN techniques including caching, de-duplication, compression, and protocol optimization. Third, the Cloud Storage Controller provides the same features and capabilities expected of local storage arrays, such as thin provisioning, automated storage tier and volume management.

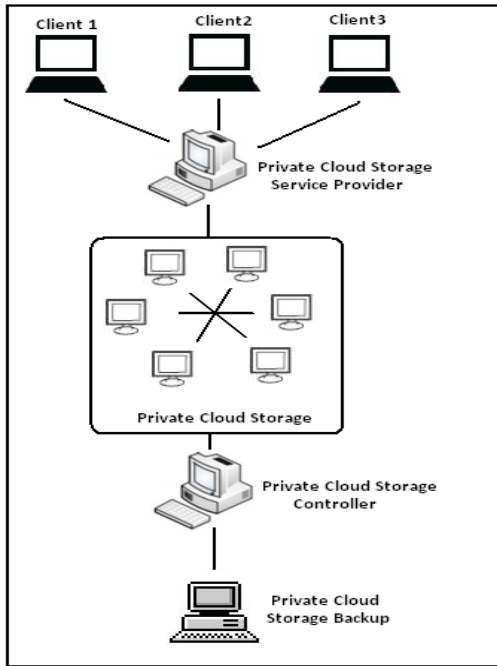


Fig. 1. Private cloud storage.

E. Overview of De-Duplication

Data De-duplication identifies the duplicate data to remove the redundancies and reduces the overall capacity of data transferred and stored. De-duplication often called as "intelligent compression" or "single-instance storage"[3] which is the method of reducing storage needs by eliminating redundant data. Only one unique instance of the data is actually retained on storage media, such as disk or tape. Redundant data is replaced with a pointer to the unique data copy. For example, if an organization webmail system might contain 50 instances of the same one megabyte (MB) file attachment. If the webmail platform is backed up or archived, all 50 instances are saved, requiring 50 MB storage space. With data de-duplication, only one instance of the attachment is actually stored. Each subsequent instance is just referenced back to the one saved copy. In this example, a 50 MB storage demand could be reduced to only one MB. Data de-duplication offers three benefits. First, lower storage space requirements will save money on disk expenditures. Second, efficient use of disk space also allows for longer disk retention periods and reduces the need for tape backups. Third, it also reduces the data that must be sent across a WAN for remote backups and replication.

F. De-Duplication Techniques

The optimization of backup storage technique is shown in Fig. 2. The Data de-duplication [4]-[6] can operate at the whole file, block (Chunk), and bit level.

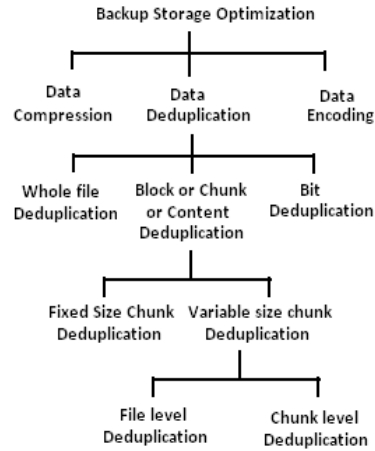


Fig. 2. De-duplication methods.

Whole file de-duplication or Single Instance Storage (SIS) [3] finds the hash value for the entire file which is the file index. If the new incoming file matches with the file index, then it is regarded as duplicate and it is made pointer to existing file index. If the new file is having new file index, then it is updated to the storage. Thus only single instance of the file is saved and subsequent copies are replaced with a pointer to the original file. **Block De-duplication** [6], [7] divides the files into fixed-size block or variable-size blocks. For **Fixed-size chunking**, a file is partitioned into fixed size chunks for example each block with 8KB or 16KB. In **Variable size chunking**, a file is partitioned into chunks of different size. Both the fixed size and variable size chunking creates unique ID for each block using a hash algorithm such as MD5 or SHA-1 [8] or MD5 [9]. The unique ID is then compared with a central index. If the ID exists, then that data block has been processed and stored before. Therefore, only a pointer to the previously stored data needs to be saved. If the ID is new, then the block is unique. The unique ID is added to the index and the unique chunk is stored. Block and **Bit de-duplication** looks within a file and saves unique iterations of each block or bit. This method makes block and bit de-duplication more efficient.

The rest of the paper is organized as follows. In Section II, we analyze the existing methods of de-duplication with its advantages and disadvantages. In Section III, we discuss about our proposed system and its functions. In Section IV, we conclude our design of DWFD and prove that our scheme greatly increases the de-duplication efficiency. We show our implementation analysis in Section V.

II. ANALYSIS OF EXISTING METHODS

In this section, we describe the advantages and disadvantages of each de-duplication methods

A. Advantages of Existing Methods

- i) Indexes for whole file de-duplication are significantly

smaller, which takes less computational time and space when duplicates are being determined. Backup performance is less affected by the de-duplication process.

ii) Fixed-size chunking is conceptually simple and fast since it requires less processing power due to the smaller index and reduced number of comparisons.

iii) In variable size chunking, the impact on the systems performing the inspection and recovery time is less. The efficiency of identifying the duplicate is high.

iv) Bit De-duplication done exact de-duplication and it is more efficient since it eliminates redundancy.

B. Disadvantages of Existing Methods

i) Whole File de-duplication is not a very efficient, because a little change within the file causes the whole file to be saved again. For example, if 500 identical attachments are sent by a insurance coordinator, this method will find all those 500 attachments that are exactly the same, but it would not find the exact duplicate copies that we have saved (i.e) Insure.Aug,Insure.Sep,Insure.Oct etc. This de-duplication checks only the size of the file regardless of the file content.

ii) In Fixed-size chunking, when a small amount of data is inserted into a file or deleted from a file, an entirely different set of chunks is generated from the updated file.

iii) The indexes for both fixed and variable size chunking are large which leads to larger index table and more number of comparisons which leads to low throughput.

iv) Bit de-duplication takes more processing time to identify the duplicate bit.

C. Methods of Block Level De-Duplication.

The block level de-duplication [10] divides the incoming file into fixed size chunks or variable size chunks. Depending on the duplicate detection of incoming chunk, the variable size chunk de-duplication can be divided into **Chunk level de-duplication** and **File level de-duplication**.

D. Chunk Level De-duplication – DDDFS

When a file has to be written, then every chunk of that file is checked for duplicate with chunks of all files. This method of detecting duplicates is **Chunk level de-duplication**. Data Domain De-duplication File System [6] DDDFS is a file system which performs chunk level de-duplication. The architecture of DDDFS is shown in Fig. 3. It supports multiple access protocols. Whenever a file to be stored, it is managed by the interfaces such as Network File System (NFS), Common Internet File System (CIFS) or Virtual Tape Library (VTL) to a generic file service layer. **File service layer** manages the file metadata using **Namespace index** and forwards the file to the content store [11]. **Content store** divides the file into variable sized chunks. Secure Hash Algorithm SHA-1 or MD5 finds the hash value for each variable size chunk, which is **ChunkID**. Content store maintains the **File Reference Index (FRI)** which contains the sequence of **ChunkID** constituting that file. **Chunk store** maintains a chunk index for duplicate chunk detection [12]. **Chunk index** is the metadata that includes **ChunkID** and the address of actual chunks in storage. Unique chunks will be compressed and stored in the container. Container is the unit of storage. In this chunk level de-duplication, the efficiency of

duplicate detection [13], [14] is high but the throughput of the de-duplication is low. So this method can be used for the applications with locality of reference between the data streams in the cloud storage.

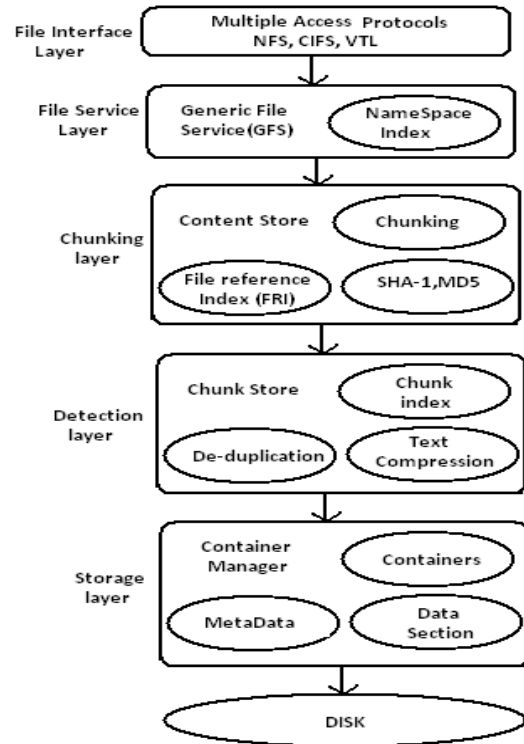


Fig. 3. Data domain de-duplication file system.

E. File Level De-Duplication – Extreme Binning

When a file has to be written, then every chunk of that file is checked for duplicate with all the chunks of the similar files. This method of detecting duplicates is **File level de-duplication**. Extreme Binning [4] uses this approach by dividing the chunk index into two tiers namely Primary index and Bin [4]. **Primary Index** contains the representative **ChunkID**, Whole file hash and pointer to bin. The disk contains **bin**, Data chunks and the File recipes. The file recipes contain the sequence of chunked for that file. Fig. 4 shows the structure of a backup node in extreme binning de-duplication. When a file has to be backed up, it performs variable size chunking and finds the representative **ChunkID** and the hash value for the entire file. The Representative **ChunkID** is checked in the primary index and if it is not there, then the incoming file is new one and a new bin is created with all **ChunkID**, chunksize and a pointer to the actual chunks are added to the disk. Then Representative **ChunkID**, file hash value and the pointer to bin of a newly created bin are added to the primary index. If the representative **ChunkID**, file hash of the incoming file is already present in the primary index, then the file is a duplicate and it is not loaded into disk and the bin is not updated. If the representative **ChunkID** of the incoming file is already present in the primary index but the hash value of the whole file does not match, then the incoming file is considered to be nearly similar to the one that is already on the disk. Most of the chunks of this file will be available in the disk [15]. The corresponding bin is loaded to RAM from the disk, and now searches for the matching chunks of the

incoming file [16]. If the ChunkID is not found in the bin, then its metadata of the chunk is added to the bin and the corresponding chunk is written to the disk. The whole file hash value is not modified in the primary index and the updated bin is written back to the disk. Here every incoming chunk is checked only against the indices of similar files [17], this approach achieves better throughput compared to the chunk level de-duplication. Since non-traditional backup workload demands better de-duplication throughput, file level de-duplication approach is more suited in this case.

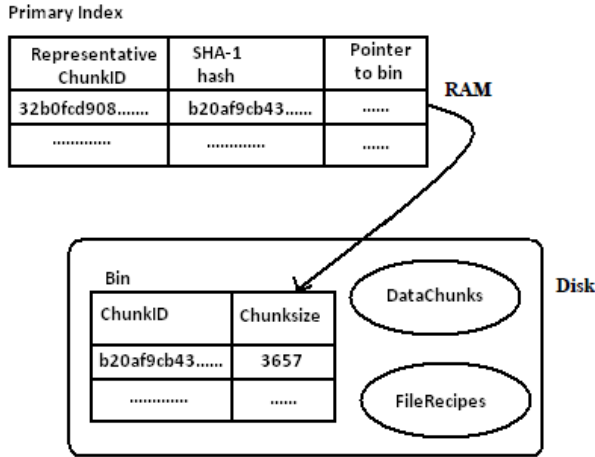


Fig. 4. Backup node in extreme binning.

III. OUR CONTRIBUTION

Cloud computing is used for better utilization of available resources [18]. The unused materials of an organization can be used to build up a private cloud. Here we try to optimize the private cloud storage backup in order to provide high throughput to the users of the organization by increasing the de-duplication efficiency.

A. Proposed System

Generally the backup of the private storage cloud belongs to the non-traditional backup. Traditional backup contains data streams with locality of reference. But the non-traditional backup contains the individual files that owns by the individual users of the organization with no locality of reference. The storage of the private cloud should be optimized as there is physical limitation on the storage space. The index of Whole File De-duplication (WFD) takes less memory space, so we try to use WFD for our private cloud storage. But the de-duplication efficiency is low in WFD due to lack of file type detection, we try to refine it further to increase the throughput and de-duplication efficiency. So we propose a new method for de-duplication namely Dynamic Whole File De-duplication (DWFD) which is the modified WFD that includes file type detection.

B. Dynamic Whole File De-Duplication (DWFD)

The existing WFD file index contains whole file hash which is used for finding the duplication regardless of the file type which leads to redundancy and it is shown in the Fig. 5. Private storage cloud consists of personal documents of the individual users belonging to organization. If we use WFD,

then there will be only one file index for all user files.

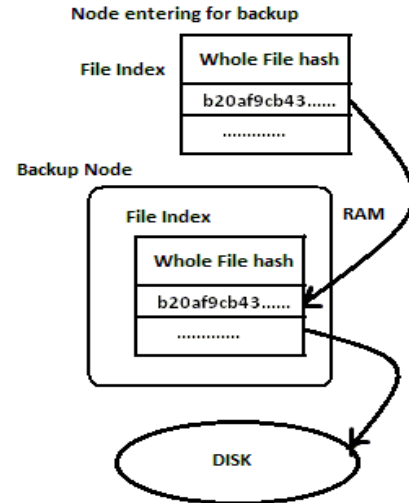


Fig. 5. Whole file de-duplication.

So all the incoming files of the different users merely waste their time for checking the single file index that reduce the throughput and de-duplication efficiency. In our Dynamic Whole File De-duplication, the users accessing the storage are identified by their unique user-id. Here we create separate file index for each user and each file belonging to an individual user is associated with the file type and is shown in Fig. 6. With this method, it is possible to group the files of each users of the organization

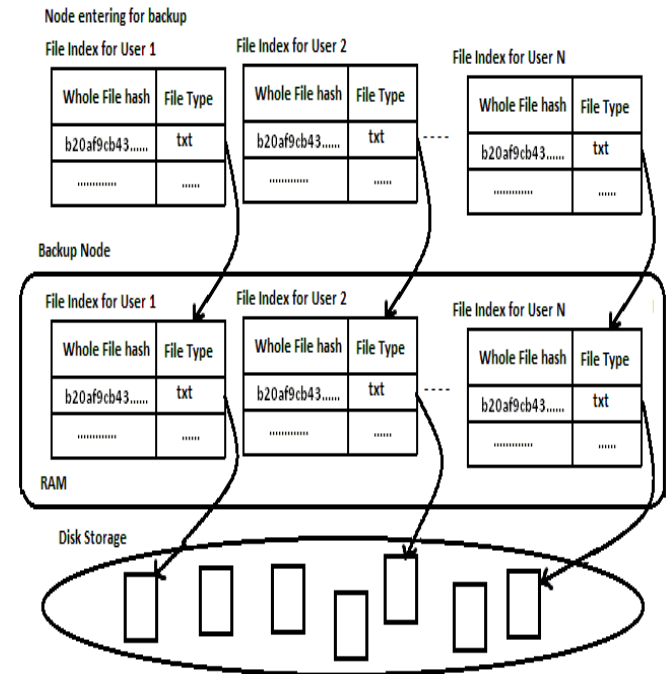


Fig. 6. Dynamic whole file de-duplication.

IV. DESIGN OF DYNAMIC WHOLE FILE DE-DUPPLICATION

Before we start our design, we have the following assumptions:

- i) The users of the private cloud are provided with separate user id. ii) The files of the individual users are collected in

separate folders in the cloud backup. Our new DWFD scheme has the following modules,

- ii) Cloud Service Providing Module
- iii) Cloud Storage Initiation Module
- iv) Cloud Storage Controller Module
- v) Cloud Backup De-duplication Module.

A. Cloud Service Providing Module

The user authentication is done in this module. If the user is new, then the registration process is done in this module and is shown in Fig. 7.

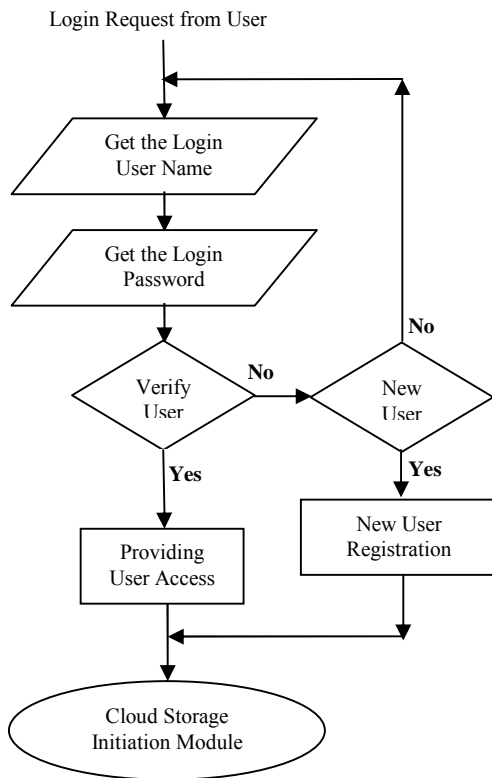


Fig. 7. Cloud Service Providing Module

B. Cloud Storage Initiation Module

After the user authentication is done in the private cloud, then he / she can start viewing, editing and saving their personal files into their folders and is shown in Fig. 8.

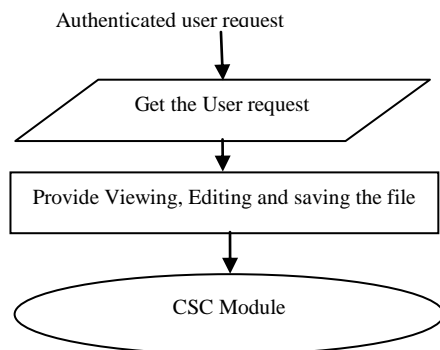


Fig. 8. Cloud storage initiation module.

C. Cloud Storage Controller Module (CSCM)

This module performs the function of integrating the files of the individual users. The file index for each user is created in this module and it is shown in Fig. 9. For each user, the

whole file hash value is found. The file type is also updated in the file index and the cloud storage controller Module is shown in Fig. 10.

File Index for User

Whole File hash	File Type
b20af9cb43.....	txt
.....

Fig. 9. File Index in DWFD

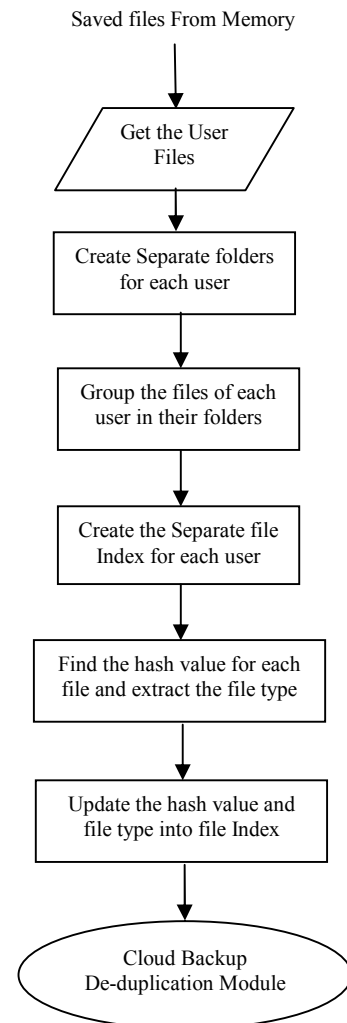


Fig. 10. cloud storage controller module.

D. Cloud Backup De-Duplication Module

This module performs the function of de-duplication detection by comparing the incoming file index with the backup node file index. It starts by checking the whole file hash. If the match is found with the hash value along with the file type, then the file is a duplicate one. If the file is identified as duplicate, then it is not saved into the disk. If the match is not found with the hash value, then the file assumed as new file and it is updated into backup node. So here the file is assumed to be duplicate if and only if both the hash value and the file type matches thereby increasing the de-duplication efficiency and it is shown in Fig. 11.

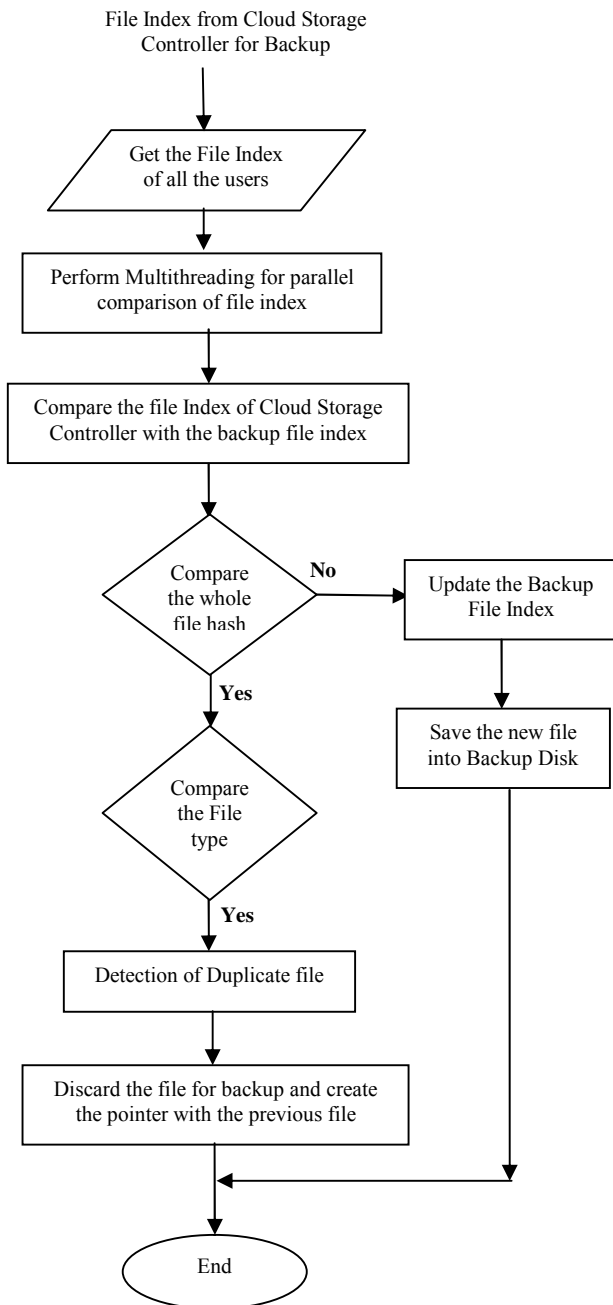


Fig. 11. Cloud backup de-duplication module.

V. IMPLEMENTATION

We have implemented this by creating the cloud server, cloud controller and multiple clients on WINDOWS platform. Any number of clients can be registered to the cloud server. The coding is done by using visual studio.Net and back end as Microsoft SQL server. The cloud server node is executed followed by the users' registration. All the users can have their individual username and password. They can upload any type of files. The class diagram is shown in the Fig. 12. Our proposed DYNAMIC WHOLE FILE DE-DUPLICATION (DWFD) is compared with the whole file de-duplication. Our analysis is showing that our proposed system will have efficiency based on the number of files being stored in the backup node. Our result analysis is shown in the Fig. 13 and Fig. 14.

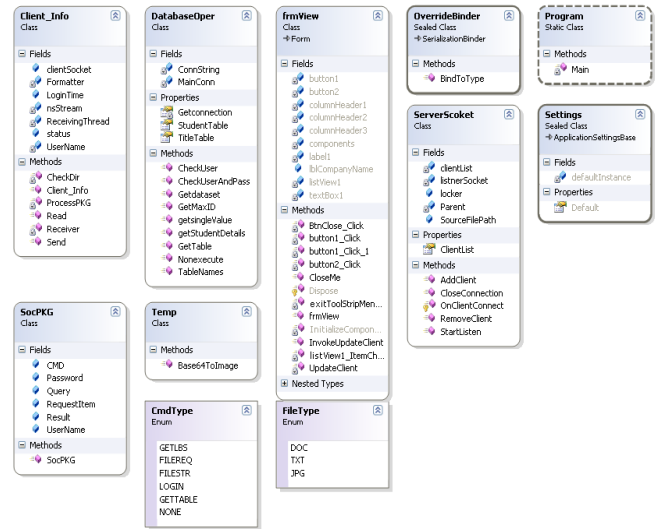


Fig. 12. Class diagram for DWFD.

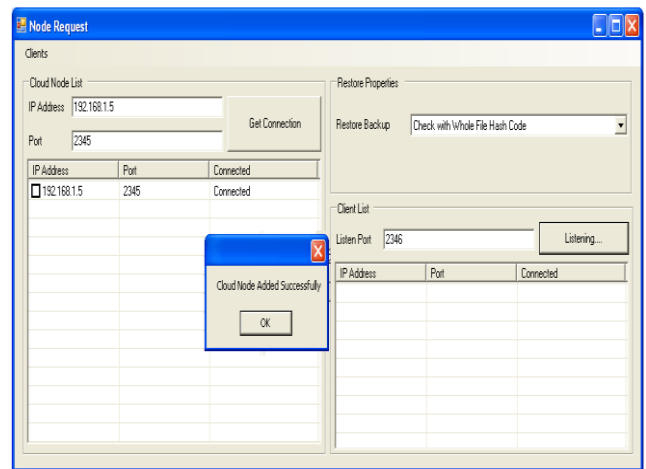


Fig. 13. Registering the client to the cloud.

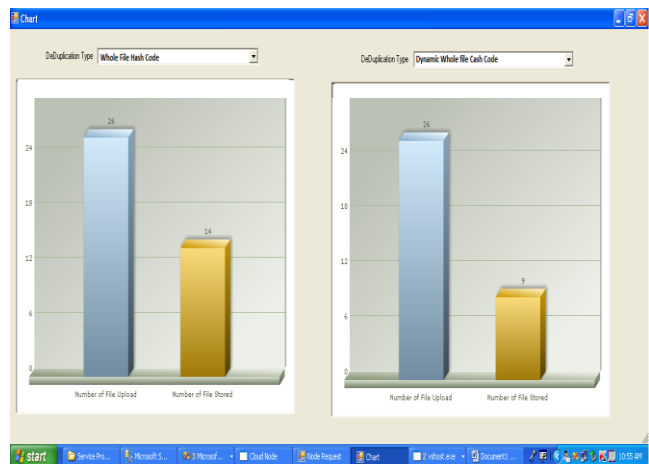


Fig. 14. Performance analysis.

VI. CONCLUSION

In this paper, we have designed our new scheme namely Dynamic Whole File De-duplication (DWFD) that effectively removes duplication. It is highly desirable to improve the private cloud backup storage efficiency by reducing the de-duplication time. Our future enhancement is to use chunk level and file level de-duplication in the private cloud storage by overcoming the negative factors in the existing Chunk level de-duplication and File level de-duplication.

REFERENCES

- [1] Y. Abe and G. Gibson, "pWalrus: Towards better integration of parallel file systems into cloud storage," in *Proc. IEEE International Conference on Cluster Computing Workshops and Posters*, pp. 1–7, 2010.
- [2] L. L. You, K. T. Pollack, and D. D. E. Long, "Deep store: An archival storage system architecture," in *Proc. Int'l Conf. Data Engineering (ICDE '05)*, pp. 804-8015, 2005.
- [3] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur, "Single instance storage in Windows 2000," in *Proc. Fourth USENIX Windows Systems Symp.*, pp. 13-24, 2000.
- [4] J. H. Min, D. Y. Yoon, and Y. J. Won, "Efficient De-duplication techniques in modern backup operation," *IEEE Transactions on Computers*, vol. 60, no. 6, June 2011.
- [5] J. S. Wei, H. Jiang, K. Zhou, and D. Feng, "Mad2: A scalable high-throughput exact de-duplication approach for network backup services," in *Proc. 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1-14, 2010.
- [6] B. Zhu, K. Li, and H. Patterson, "Avoiding the disk bottleneck in the data domain de-duplication file system," in *Proc. the 6th USENIX Conference on File and Storage Technologies, FAST'08*, Berkeley, CA, USA: USENIX Association, vol. 18, pp. 1–14, 2008.
- [7] D. Bhagwat, K. Eshghi, D. D. E. Long, and M. Lillibridge, "Extreme binning: Scalable, parallel de-duplication for chunk-based file backup," *MASCOTS*, pp. 1-9, 2009.
- [8] National Institute of Standards and Technology. (Apr. 1995). Secure hash standard. FIPS 180-1. [Online]. Available: <http://www.itl.nist.gov/fipspubs/fip180-1.htm>
- [9] R. Rivest. (Apr. 1992). The MD5 message-digest algorithm. IETF, Request for Comments (RFC) 1321. [Online]. Available: <http://www.ietf.org/rfc/rfc1321.txt>
- [10] M. Lillibridge, K. Eshghi, D. Bhagwat, V. Deolalikar, G. Trezise, and P. Camble, "Sparse indexing: Large scale, inline deduplication using sampling and locality," in *Proc. Seventh USENIX Conf. File and Storage Technologies (FAST'09)*, 2009.
- [11] C. Polcironiades and I. Pratt, "Alternatives for detecting redundancy in storage systems data," in *Proc. the Annual Conference on USENIX Annual Technical Conference*, pp. 1-15, 2004.
- [12] A. Z. Broder, "On the resemblance and containment of documents," in *Proc. the Compression and Complexity of Sequences*, 1997, pp. 21–29.
- [13] P. Kulkarni, F. Dougliis, J. LaVoie, and J. Tracey, "Redundancy elimination within large collections of files," in *Proc. USENIX Ann. Technical Conf., General Track*, pp. 59-72, 2004.
- [14] B. Hong and D. D. E. Long, "Duplicate data elimination in a san file system," in *Proc. 21st IEEE / 12th NASA Goddard Conf. Mass Storage Systems and Technologies (MSST)*, pp. 301-314, Apr. 2004.
- [15] C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki, "HYDRAsTOR: A scalable secondary storage," in *Proc. the 7th USENIX Conference on File and Storage Technologies (FAST)*, San Francisco, CA, USA, Feb. 2009.
- [16] C. Polcironiades and I. Pratt, "Alternatives for detecting redundancy in storage systems data," in *Proc. Conf. USENIX '04*, June 2004.
- [17] D. R. Bobbarjung, S. Jagannathan, and C. Dubnicki, "Improving duplicate elimination in storage systems," *ACM Trans. Storage*, vol. 2, no. 4, pp. 424-448, 2006.
- [18] W. Zeng, Y. Zhao, K. Ou, and W. Song, "Research on cloud storage architecture and key technologies," in *Proc. the second International Conference on Interaction Sciences*, pp. 1044-1048, 2009.



M. Shyamala Devi was born in Madurai in 1984. She has completed B.E in computer science and engineering at P.S.N.A College of Engineering and Technology, Dindigul, TN, India in 2005. She completed her M.E in computer science and engineering at P.S.N.A College of Engineering and Technology, Dindigul, TN, India in 2009. She Completed her M.B.A systems area at Madurai Kamaraj University, Madurai, TN, India. She is now Pursuing Ph.D in Anna University, Chennai.

She worked as a lecturer at P.S.N.A College of Engineering and Technology, Dindigul, TN, India from 2005 to 2009. Then she joined as an assistant professor at R.M.D Engineering College, Chennai, TN, India from 2009 to till date. She has authored 7 engineering books titled Theory of Computation, Principles of Compiler Design, Data Structures and Algorithm Analysis, Graphics and Multimedia, Fundamentals of Computer Programming, Digital Computer Fundamentals and Visual Programming, by Shri Krishna HiTech Publishing Pvt Ltd, Chennai, TN, India. She have presented paper in three IEEE international Conference like IACC2009 at Thapar University, Patiala, Punjab on March 6 to 7, 2009, ICCCN 2008 at Karur, TN, India on Dec 18 to 20, 2008 and ADCOM 2006 at NIT, Surathkal, Karnataka, India on Dec 20 to 23, 2006. She have published three papers in IEEEExplore titled 'Dynamically Distributed Parallel Periodic Switching (D2PS) for minimizing file download time in Peer-to-Peer Networks', 'Dynamically Distributed Parallel Permanent Connection (D2P) for minimizing file download time in Peer-to-Peer Networks' and 'Security On-Demand Position Based Private Routing Protocol for Mobile Adhoc Network'

Mrs. M. Shyamala Devi is an active life member of CSI, ISTE, ICST, IAEST and IACSIT. She has received a funded project from CSI on March 2010 for 'Web based speech recognition for visually challenged Users'.



V. Vimal Khanna was born in Chennai in the year 1994. He has completed his SSLC at Sri Ram Dhayal Vivekananda Vidhyalaya Junior College, Chennai TamilNadu, India in the year April 2009. He completed his HSC at Vidya Vikas Higher Secondary School, Erode, TamilNadu, India in the year April 2011. Currently he is Third year student pursuing B.E in computer science and engineering at R.M.D Engineering College, Chennai, TamilNadu, India. He is an active student member

of CSI. He is online certified professional in artificial intelligence. He is an active student member of CSI. He is a BEC certified Professional. He is a member of Entrepreneurial Development Cell of R.M.D Engineering college, Chennai, TamilNadu, India



A. Naveen Bhalaji was born in vellore in the year 1994. He has completed his SSLC at Cross Matric Higher Secondary School, TamilNadu, India in the year April 2009. He completed his HSC at Sunbearn Matric Higher Secondary School, TamilNadu, India in the year april 2011. Currently he is a third year student pursuing B.E in computer science and engineering at R.M.D Engineering College, Chennai, TamilNadu, India. He have done various certification in C , C++, (A+) in networking from

NIIT, Chennai, Tamilnadu, India. He is an active student member of CSI.