International Journal of Engineering Research in Computer Science and Engineering Engineering (IJERCSE) Vol 2, Issue 3, March 2015

Implementation of Source Deduplication For Cloud Backup Services By Exploiting Application Awareness

[1]Nehal Markandeya,[2]Sandip Khillare, [3]Rekha Bagate,[4]Sayali Badave ,[5]Vaishali, Barkade
[1][2][3]Department of Computer Engineering,JSPM's Rajarshi Shahu College of Engineering
Pune, India

Abstract — **In personal computing devices that rely on a cloud storage environment for data backup, an imminent challenge facing source deduplication for cloud backup services is the low deduplication efficiency due to a combination of the resource intensive nature of deduplication and the limited system resources. In this paper, we present ALG-Dedupe, an Application-aware Local-Global source deduplication scheme that improves data deduplication efficiency by exploiting application awareness, and further combines local and global duplicate detection to strike a good balance between cloud storage capacity saving and deduplication time reduction. The storage service provider has to store data in the form of chunks, it is first generates fingerprints by using hash functions and transferred over WAN.**

Key words- **application awareness, cloud backup, personal storage, source deduplication.**

I. INTRODUCTION

Now days the ever-growing volume and value of digital information have raised a critical and increasing requirement for data protection in the personal computing environment. Cloud backup service has become a cost-effective choice for data protection of personal computing devices, since the centralized cloud management has created an efficiency and cost inflection point, and offers simple offsite storage for disaster recovery, which is always a critical concern for data backup. The efficiency of IT resources in the cloud can be further improved due to the high data redundancy in backup dataset [1].

Data deduplication is an effective compression approach for data redundancy. The large object is divided into small partition called as chunk. Chunk represents fingerprints. Fingerprint is hash value of that chunk. The duplicated chunks replaced with their fingerprint after the chunk fingerprint index lookup and only transfers and store unique data chunk for purpose of storage efficiency. However, data deduplication is a resource-intensive process, which entails the CPU-intensive hash calculations for chunking and fingerprinting and the I/O-intensive operations for identifying and eliminating duplicate data. Unfortunately, such resources are limited in a typical personal computing device. Therefore, it is desirable to achieve a tradeoff (i.e., deduplication efficiency) between deduplication effectiveness (i.e., duplicate elimination ratio) and system overhead for personal computing devices with limited system resources [1].

I. EXISTING SYSTEM

The existing deduplication strategy is divided into two parts: Local source deduplication detect redundancy in backup dataset from the same device on client side and only sends the unique data chunks to the cloud storage. Global source deduplication performs duplicate check in backup datasets from all clients in the cloud side before data transfer over WAN [3].

The former only eliminates intra-client redundancy with low duplicate elimination ratio by low-latency client-side duplicate data check, while the latter can suppress both intra-client and inter-client redundancy with high deduplication effectiveness by performing high-latency duplication detection on the cloud side [4]. Inspired by Cloud4Home that enhances data services by combining limited local resources with low latency and powerful Internet resources with high latency, local-global source deduplication scheme that eliminates intra-client redundancy at client before suppression inter-client redundancy in the cloud, can

potentially improve deduplication efficiency in cloud backup services to save as much cloud storage space as the global method but at as low latency as the local mechanism [3].

II. PROPOSED SYSTEM

We propose ALG-Dedupe, an Applicationaware Local-Global source deduplication scheme that not only exploits application awareness, but also combines local and global duplication detection, to achieve high deduplication efficiency by reducing the deduplication latency to as low as the application-aware local deduplication while saving as much cloud storage cost as the application-aware global deduplication. Our application-aware deduplication design is motivated by the systematic deduplication analysis on personal storage. We observe that there is a significant difference among different types of applications in the personal computing environment in terms of data redundancy, sensitivity to different chunking methods, and independence in the deduplication process [1].

Thus, the basic idea of ALG-Dedupe is to effectively exploit this application difference and awareness by treating different types of applications independently and adaptively during the local and global duplicate check processes to significantly improve the deduplication efficiency and reduce the system overhead. Plus as part of adding something new to it we propose cryptographic encryption of chunks saved at cloud storage. The aim and objective of paper is that to employ an intelligent data chunking method, adaptive use of hash function based on application awareness and to create application-specific indices in an application aware index structure.

III. DEDUPLICATION SCHEMES

Another way to think about data deduplication is by where it occurs. When the deduplication occurs close to where data is created, it is often referred to as "source deduplication." When it occurs near where the data is stored, it is commonly called "target deduplication"[3].

Source deduplication: It ensures that data on the data source is deduplicated. This generally takes place directly within a file system. The file system will periodically scan new files creating hashes and compare them to hashes of existing files. When files with same hashes are found then the file copy is removed and the new file points to the old file. Unlike hard links however, duplicated files are considered to be separate entities and if one of the duplicated files is later modified, then using a system called Copy-on-write a copy of that file or changed block is created. The deduplication process is transparent to the users and backup applications. Backing up a deduplicated file system will often cause duplication to occur resulting in the backups being bigger than the source data.

Target deduplication: It is the process of removing duplicates of data in the secondary store. Generally this will be a backup store such as a data repository or a virtual tape library [3].

V. MODULES

5.1. File Size Filter: This module checks if uploaded file is less than the defined chunk sizes. If less it won't send it for chunking and put them in different segment store. Once a segment is full, the file is chunked.

5.2. Intelligent Chunker: This module divide files into three categories depending on file type i.e. Compressed files, static uncompressed files and dynamic uncompressed files. Static uncompressed files are not editable. These files chunked into fixed size by using SC chunking like file with extension .pdf, .exe, etc. Dynamic uncompressed files are editable. It breaks dynamic uncompressed files variable-sized chunks using CDC based chunking like file format .txt, .doc.

5.3. Application-Aware Deduplicator: After data chunking in intelligent chunker module, data chunks will be deduplicated in the application aware deduplicator by generating chunk fingerprints in the hash engine and detecting duplicate chunks in both the local client and remote cloud. If static chunking then SHA1 is used for fingerprinting. If CDC then MD5 is used for generating fingerprint. To achieve high deduplication efficiency, the application aware deduplicator first detects duplicate data in application aware local index corresponding to local dataset. And then compares local deduplicated data chunks with all stored cloud dataset looking up fingerprints in application aware global index on cloud side. After that only unique data chunks will be stored in cloud storage with parallel container management.

5.4. Application-Aware Index Structure: It consist of small hash table and application index based on disk indices classified by file type. Index structure is used for speed up the I/O operation. It uses two application aware indices: local index on client side and global index on cloud side. It can achieve the high deduplication throughput by looking up chunk fingerprints concurrently in small indices classified by application type.

5.5. Segment and Container Management: The segment contains the files which is less than predefined size. ALG-Dedupe often group deduplicated data from many files and chunk into larger units called segments before these data transferred over WAN.

After a segment is sent to the cloud, it will be routed to a storage node in the cloud with its corresponding fingerprints, and be packed into container, a data stream based structure, to keep spatial locality for deduplicated data. A container includes a large number of chunks and their metadata, and it has a size of several MB. An open chunk container is maintained for each incoming backup data stream in storage nodes, appending each new chunk or tiny file to the open container corresponding to the stream it is part of. When a container fills up with a predefined fixed size, a new one is opened up. If a container is not full but needs to be written to disk, it is padded out to its full size. This process uses chunk locality to group chunks likely to be retrieved together so that the data restoration performance will be reasonably good. Supporting deletion of files requires an additional process in the background.

VI. ARCHITECTURE OVERVIEW

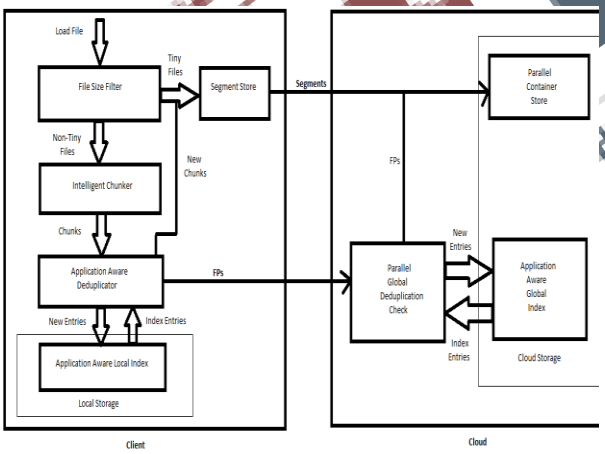

Fig 1.Architecture Overview of ALG-Dedupe Design

An architectural overview of ALG-Dedupe is illustrated in Fig. 1, where tiny files are first filtered out by file size filter for efficiency reasons, and backup data streams are broken into chunks by an intelligent chunker using an application aware chunking strategy. Data chunks from the same type of files are then deduplicated in the application-aware deduplicator by generating chunk fingerprints in hash engine and performing data redundancy check in application-aware indices in both local client and remote cloud. Their fingerprints are first looked up in an application-aware local index that is stored in the local disk for local redundancy check. If a match is found, the metadata for the file containing that chunk is updated to point to the location of the existing chunk. When there is no match, the fingerprint will be sent to the cloud for further parallel global duplication check on an application-aware global index, and then if a match is found in the cloud, the corresponding file metadata is updated for duplicate chunks, or else the chunk is new. On the client side, fingerprints will be transferred in batch and new data chunks will be packed into large units called segments in the segment store module with tiny files before their transfers to reduce cloud computing latency and improve network bandwidth efficiency over WAN. On the cloud data center side, segments and its corresponding chunk fingerprints are stored in a self describing data structure container in cloud storage, supported by the parallel container store.

FUTURE WORK

As direction of future work, we plan to further optimize our scheme for other resource-constrained devices like smartphone or tablet and investigate secure deduplication issue in cloud backup services of personal computing environment.

REFERENCES

[1] Yinjin Fu, Hong Jiang, Senior Member, IEEE, Nong Xiao, Member, IEEE, Lei Tian, Fang Liu, and Lei Xu "Application-Aware Local-Global Source Deduplication for Cloud Backup Services of Personal Storage" IEEE Transactions on parallel and Distributed Systems, Vol.25, No.5, May 2014.

[2] Y. Fu, H. Jiang, N. Xiao, L. Tian, and F. Liu, ''AA-Approach for Cloud Backup Services in the Personal Computing Environment,'' in Proc. 13th IEEE Int'l Conf. CLUSTER Computing, 2011, pp. 112-120

[3] www.wikipedia.com

[4] S. Kannan, A. Gavrilovska, and K. Schwan, ''Cloud4Home- Enhancing Data Services with @Home Clouds,'' in Proc. 31st ICDCS, 2011, pp. 539-548.