ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

An Enhanced Technique for Deduplication Using Cloud Computing

Ms.Yamuna.M Dr.Srinivasan.S M.E. Ph.D.

Pg Scholar, Computer Science and Engineering Department, Anna University Regional Campus, Keelakuilkudi, Madurai-625019 INDIA; E-Mail:yamunacutie93@gmail.com).
Head oftheDepartment, Computer Science and Engineering Department, Anna University Regional Campus, Keelakuilkudi, Madurai-625019 INDIA; E-Mail:sriniss@yahoo.in).

Abstract: Data Deduplication is an emerging technique in storage systems. It includes a space-efficient approach for data backup and archiving. The proposed work expresses the novel Deduplication architecture called I-sieve and it is applied in a slow pace for inline scenarios such as virtual machines and cloud storage systems. As an essential technology in cloud computing, VM has been a blistering research issue in recent times. In this system the architecture for inline Deduplication is based on existing File System protocol which addresses performance challenges for primary storage. The design corresponding index as well as the mapping table presents a multi-level cache using a solid state drive to reduce RAM consumption and optimize lookup performance. In Solid State Drives (SSDs) the issues are price per GB of storage space, and issue of writing limit or disk endurance. Special chunking algorithm such as fixed, variable and content aware chunking methods are used to decide the chunk size of Deduplication. The evaluation results show excellent Deduplication and backup systems.

Keywords-de-duplication; I-sieve; SSD; inline; blistering; chunking.

1. INTRODUCTION

Data Deduplication is one of the hottest technologies for improving storage efficiency, because it enables companies to save a lot of money on storage costs to store the data. This is great news for cloud providers, because if we store less, we need less hardware. If we can de-duplicate what we store, we can better utilize existing storage space, which can save money by using what we have more efficiently. If we store less, we also send less data over network. It saves money in hardware and network costs over time. Data de-duplication can significantly reduce backup data storage, reducing storage capacity, space and energy consumption. No de-duplication systems exist that de-duplicate in line with client requests for latency sensitive primary workloads. All prior de-duplication work focuses on either: throughput sensitive archival and backup systems or ii) latency sensitive primary systems that de-duplicate data offline during idle time or iii) file systems with inline Deduplication, but hesitant to performance. In order to overcome the drawback, this paper introduces two novel insights that enable latency-aware inline, primary de-duplication. The challenge of inline de-duplication is to not increase the latency, but increases the foreground operations. Reads are affected by the fragmentation in data layout that naturally occurs when de-duplicating blocks across many disks [5].

With the potentially unlimited storage space offered by cloud earners, the operators inclined to use a huge amount of space as they can and vendors continually look for techniques aimed to reduce redundant data and exploit space savings. A technique which has been widely accepted is cross-user de-duplication. The modest impression behind de-duplication is to accumulate duplicate data, if a user wants to upload a file (block) which is already stored, the cloud operators will insert the user to the owner list of that file (block). De-duplication has demonstrated to grasp high space and cost savings and many cloud storage operators are presently implement it. De-duplication excludes surplus data parts from the backup and reduces the size of the backup data [6].De-duplication can be enabled for derivative copies during Storage Policy creation. The non-de-duplicated data can be de-duplicated by enabling Deduplication on the secondary copy. The de-duplication process as shown in fig1.

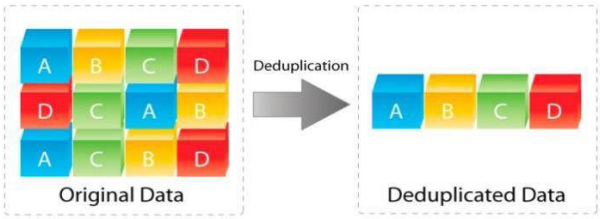

Fig 1: Deduplication process

Chunk-based Deduplication, a popular de-duplication technique, first divides input data streams into fixed or variable-length chunks. Typical chunk sizes are 4 to 8 kobo. A cryptographic hash or chunk ID of each chunk is used to determine if that chunk has been backed up before. Chunks with the same chunk ID are assumed identical [9].

In summary, our key contributions primarily include the following.

_ Analysis of the features of current Deduplication systems together with a detailed summary of their challenges.

_ an inline Deduplication system (I-sieve), for primary workloads, that serves as a module embedded in the cloud storage system.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

_ Design of novel index tables aimed at block level Deduplication and a new multi-cache structure using SSD to boost foreground performance.
_ Implementation of an I-sieve prototype iscsi target.
_ Evaluation of our I-sieve performance by
Pretending real world presentations such as
Office and VM environments.

II RELATED WORK

In traditional de-duplication system there are two major input and output operations such as read and write. The I/O requests from the front end, data de-duplication operation i.e. is happened on the write path of the file system. It can be carried out in four steps:

A. FILE CHUNKING:

In this step, all files are divided into different types of blocks by the chunking algorithms. The system then computes a fingerprint of each function such as SHA-1 and MD5.

*B.*DE-DUPLICATION CHECKING:

The fingerprint is checked against the fingerprint table. If the block is a duplicated copy it may not written into the disk for that the Meta data is recorded into the disk. If the block does not have any duplicate copies, then the fingerprint of this block is recorded as a unique block into the system.

C.META DATA UPDATING:

The block indexing process which is completed then the corresponding metadata of the file is updated .Similarly the metadata is stored in designated zones on the disks by periodic operations.

*D.*DATA BLOCK STORAGE:

The final step in the right path is to store the unique data blocks. The storage elements are organized by containers or bins, which can be specified by number of data blocks on the disk.

CLASSIFICATION OF DE-DUPLICATION SYSTEMS:

Currently there are many classifications for de-duplication systems.

*A.*PRIMARY VERSUS SECONDARY DE-DUPLICATION SYSTEMS:

Primary Deduplication systems are designed for improving performance, in which workloads are sensitive to I/O latency. Secondary Deduplication systems are primarily used for secondary storage environments, such as backup applications, which require high data throughput.

*B.*POST PROCESSING VERSUS INLINE DE-DUPLICATION*:*

Post processing Deduplication is an out-of-band approach where data is not de-duplicated until after the backup has completed. On the other hand, with inline Deduplication, chunk hash calculations are created on the target device as data enters the storage devices in real-time. In this paper, we primarily focus on the inline Deduplication system for its high space utilization and real-time characteristics.

The current Deduplication studies, and lists the following observations that drive us to build a new Deduplication system.

- More duplicate data exists in many typical environments.
- Multipart data management in the work path.
- Poor performance in inline Deduplication systems. Most of the current de-duplication systems are used with backup scenarios. However, for real-time applications, these systems are not suitable due to their read or write performance.

Deduplication technology has been widely used in most inline storage systems including primary [6] and secondary storage. There are also some differences in Deduplication granularity. In file level Deduplication [10], the Deduplication ratio is not obvious [13] when compared with block level Deduplication [4–7] or sub-file level [8–11]. Therefore, block chunking algorithms are often used to realize block level Deduplication, including fixed-sized and variable-sized algorithms. Some applications (e.g., office, virtual machine environments, etc.) are well suited for fixed-sized Deduplication. To improve foreground performance for inline Deduplication, dialup [6], a latency-aware inline data Deduplication system for primary storage describes a novel mechanism to reduce latency. It takes advantage of the spatial locality and temporal nature of the primary workloads in order to avoid extra disk I/Os and seeks.iDedup is complementary to I-sieve, and together they can further shorten response time. Besides the optimization based on workloads, Chunk-Stash [7] uses a three-tier Deduplication architecture by adding a fast device, SSD. By utilizing SSD, the penalty of index lookup misses in RAM is reduced by orders of magnitude because such lookups are served from a fast, flash-based index. In the same spirit, we also employ SSD as a strategy for improving performance in I-sieve. In contrast to current optimizations for inline Deduplication systems, I-sieve improves performance because of a shorter Deduplication operation path and multi-cache structure. I-sieve can serve as an I/O filter between file I/O and disk I/O. There is no processing logic for data, such as “container” or “bin” structures in other approaches.

We propose an inline, block level primary Deduplication system with high performance called I-sieve. The goal of I-sieve is to build a small-scale storage system for use in an office or private-cloud environments. In order to address the challenge of low performance in inline Deduplication systems, a lightweight indexing table and a two-level cache structure are used to improve de-duplication performance in I-sieve. The evaluations show excellent trade-offs of I-sieve between foreground performance and the de-duplication fraction.

III PROPOSED SYSTEM

Our I-sieve system always assumes that the total size of the storage system is 10 TB; all designs of the

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

mapping table and cache are based on this assumption. I-sieve is a backend storage module and all the foreground clients access the storage server using the iscsi protocol. Thus, I-sieve can be perceived as a data sieve, which eliminates much of the data redundancy seen in file systems. The implementation of I-sieve is a module imbedded into iscsi; therefore, it can be incorporated with many storage applications and also be easily deployed.

I-sieve consists of three key functional components, Deduplication engine, Multi-cache and a Snapshot module.

The Deduplication engine provides block-level data Deduplication, including hash table (also called fingerprint table) management and an I/O remapping module. All the foreground write requests are handled within the Deduplication engine, and then transferred to disk request queues.

Data de-duplication technology to use mathematics for each data element, "hash" algorithms to deal with, and get a unique code called a hash authentication number. Each number is compiled into a list; this list is often referred to as hash index. At present mainly the file level and block-level, they can be optimized for storage capacity.

A. FILE-LEVEL:
File-level Deduplication is often referred to as Single Instance Storage (SIS), check the index back up or archive files need the attributes stored in the file with the comparison .If not the same file, it will store and update the index; Otherwise, the only deposit pointer to an existing file. Therefore, the same file saved only one instance, and then copy all the "stub" alternative, while the "stub" pointing to the original file.

B. BLOCK-LEVEL:
Block-level data Deduplication technology to data
stream divided into blocks, check the data block, and determine whether it met the same data before the block .If the block is unique and was written to disk, its identifier is also stored in the index; Otherwise, the only deposit pointer to store the same data block's original location. This method terms with a small-capacity alternative to the duplication of data blocks, rather than storing duplicate data blocks in the disk storage space. Hash algorithm used to judge duplicate data, may lead to conflict between the hash error. MD5, SHA-1 hash algorithm are checked against the data blocks to form a unique code.
The Multi-cache module primarily handles cache requests, and also acts as a bridge that connects memory, SSD, and disk devices. The snapshot module is responsible for the reliability of the de-duplicated data blocks; it does so by performing periodic snapshot operations.

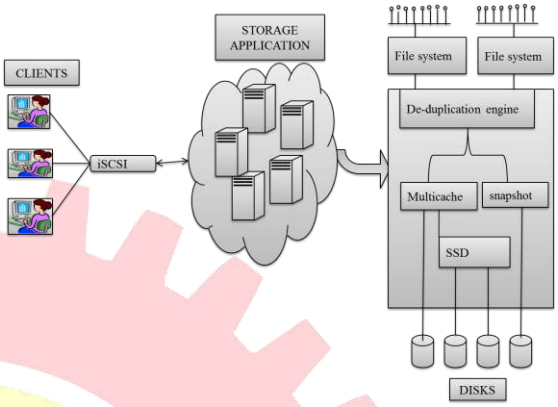

Fig 2 Architectural overview of I-sieve

DESIGN APPLICATIONS:

A.THE DESIGN OF INDEX TABLES:

In Fig. 2, the index tables, including the fingerprint table and other mapping tables, are required in every Deduplication operation. The efficient design of the index table can result in great performance improvements, especially for the inline primary Deduplication system. There are two challenges to be addressed in the design of the fingerprint index tables in I-sieve, low cost memory consumption and efficient remapping logic. Generally, the traditional fingerprint table uses the full hash index for data Deduplication; however, this can result in memory exhaustion due to large data volumes.

B.MULTI-LEVEL CACHE OF I-SIEVE:
The optimization of index performance, the organization of the index tables and data blocks is important for the overall system performance as well. Therefore, we introduce a flash-based storage device SSD acting as a fast cache between RAM memory and disk.
I-sieve uses a Cache Mapping Table (CMT) to manage different metadata and data caches in RAM and SSD. For the metadata, I-sieve uses two-level mappings, RAM-SSD, which means that the SSD stores all the metadata. Some of frequently-used metadata is migrated from SSD or disk to RAM in order to boost performance. Specifically, the first three levels of buckets are persisted in RAM, only taking up about 64 MB. Due to the scale of each HT shown in Fig. 3, I-sieve selects some hot HTs based on request count to store in RAM memory; the rest are stored on SSD. For the cached data blocks, there are three tiers of mapping in I-sieve, RAM, SSD, and Disk. All de-duplicated data blocks are eventually stored on disks; however, newly added blocks are written to temporary SSD locations and then imported into disks in a particular time zone. The system also caches hot spot blocks in RAM in order to improve read performance; the selection of blocks is based on the value of the reference count described in HT.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

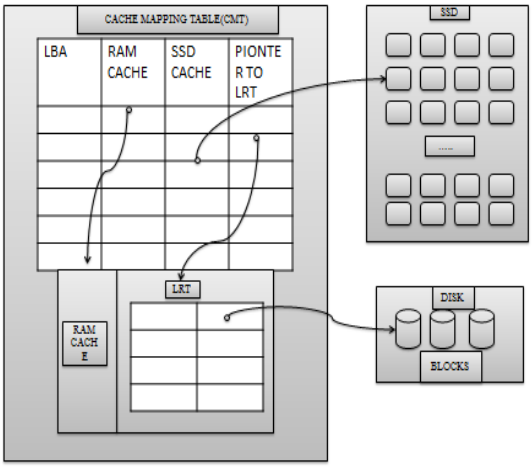

Fig 3 Structure of Multicache.

IV IMPLEMENTATION:

In business systems, the duplication has occurred the implementation can be divided into foreground and background processing two types of treatment.

Foreground processing is done through means of pure software implementation. Software itself is a rub out function of the backup software, customer/machine oriented structure. Server to develop strategies and initiate a backup at the appointed time, scheduling Client on the backup data block, data block input Hash operation procedures, the results included in Hash Table Block behind the operation result with the same value exists in the table to rub out the data, record and save the different results on the block. Data recovery and preservation of the value which under the table of data blocks to restore the deleted data, data recovery into restructure the data. Implementations took an advantage of data output to the network before the hosts in the business to achieve the rub out operation, to reduce network traffic and reduce the amount of storage space.

Background processing method used to achieve integrated software and hardware equipment, the overall advantage of the heavy equipment used to delete the CPU and memory, the operation of the business applications are not affected. Deletion in the integrated device in accordance with re-location took place is divided into In-line and Post-processing implemented in two ways.

In-line implementations also based on the Hash algorithm, data written to device memory, delete the buffer for re-operation, variable-length data block partition function, it first scans data, and analysis can produce the extreme repetition. The rate of the split point that splits the data and form variable size data blocks, after rub out process based on Hash algorithm principle.

Post-processing methods used to achieve differential algorithm Hash algorithm technology or technology. The biggest difference with other implementations when the data is written, does not deal with being directly saved to the integration of storage devices, and then to re-delete operation. Based on 1 Byte units can scan the maximum found in the duplication of data, provides a maximum deletion ratio, according to dozens of different data types can achieve a ratio of 1 to 1000 than the compression utility. As the latest data is so well preserved that in accordance with the rules and regulations for the need to ensure the authenticity of the user data copy, meet compliance requirements.

The basic steps to de-duplicate data consist of five stages:

The first phase of data is the collection phase, by comparing the old and the new backup data backup, reducing the scope of the data.

Comparing the second phase of the process of identifying data, in bytes, of the data collection phase marks a similar data objects. If the first phase of the work sheet created the need for data identification, then the usage of specific algorithm to determine which data backup group is unique, what data is repeated. If the first phase identified from the meta-data level of data and backup group, the same as the previous backup, then the recognition stage in the data bytes of data will be compared.

The third phase of the data is re-assembled, new data is saved, the previous stage was marked duplicate data is saved data pointer replacement. The end result of this process is to produce a copy of the deleted after the backup group view.

The fourth stage will actually remove all the duplicate data before performing a data integrity check efficacy. Finally remove the redundant storage of data, the release of previously occupied disk space for other uses.

In contrast to current optimizations for inline Deduplication systems, I-sieve improves performance because of a shorter Deduplication operation path and multi-cache structure. I-sieve can serve as an I/O filter between file I/O and disk I/O. There is no processing logic for data, such as ample or box structures in other approaches.

V EXPERIMENTAL RESULTS:

The results for Deduplication checking that shows the excellence foreground and background performance. The de-duplication is one the process for implementing without duplicate entries which is written in to the disk. De-duplication is not only check the files and also the disk, websites etc.., Duplicate data exists in the form of duplicate virtual machine image files, attachments to e-mails, duplicate copies of files, etc., all of which increases the loading track. De-duplication technologies pursue to identify and eliminate duplicate copies of data at the file or block level. Similarly, data can also be compressed to achieve space efficiency.for example, if there are thousands of files in the record the single person cannot check all those files.Soin order to check all files in the record. We use Deduplication process. If the block has a duplicate in the table, then this block is not written to disk. Instead only the Meta data is recorded. However, if the block does not have a duplicate in the table then the fingerprint of this block is inserted into fingerprint table as a unique block in the system. This step involves an extra fingerprint indexing process to be performed to check for duplicate data when compared with the traditional I/O

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

operations. In this process the thousand files are selected and kept in the engine. That type of engine is known as de-duplication engine. In that engines the files that contain same content that are present in the files which are kept in the arrays. If there is a duplicate enters in the files which are kept in the index tables. Each of the word present in the files that have their own id generation i.e. the id's generation is separate for duplicate and de-duplicates entries. Here, the multi-cache module works for the reliability of the de-duplicate copies. This module is present in-between the de-duplication engine module and the snapshot module.As the block indexing process completes, the corresponding metadata of the file is updated, including the mapping of blocks and files based on pointers. Subsequently, the metadata is stored in designated zones on the disk by periodic dosage operations. Cache memory also called CPU memory is random access memory (RAM) that a computer microprocessor can access more quickly than it can access regular RAM. This memory integrated straight with the CPU chip or placed on a separate chip that has a separate bus interconnect with the CPU. The basic purpose of cache memory is to store program instructions that are frequently re-referenced by software during operation. Fast access to these instructions increases the overall speed of the software program. The basic purpose of cache memory is to store program instructions that are frequently re-referenced by software throughout all processes. Fast contact to these instructions increases the overall speed of the software program. The final step in the right path is to store the unique data blocks. The storage elements in most current system are organized by containers or bins which contain a specified number of data blocks on the disk. When the container or bin reaches the target size, it is sealed and written systems. A repository can contain multiple snapshots of the same cluster; these snapshots are identified by unique names within the cluster. By default a snapshot of all open and started indices in the clusters are created. This behavior can be different by requiring the list of indices in the body of the snapshot request. Besides creating a copy of each index the snapshot process can also store universal cluster metadata, which includes persistent cluster settings and templates. The transient settings and registered snapshot repositories are not stored as part of the snapshot. Only one snapshot process can be executed in the cluster at any time. So, the results shows that an excellent de-duplication data.

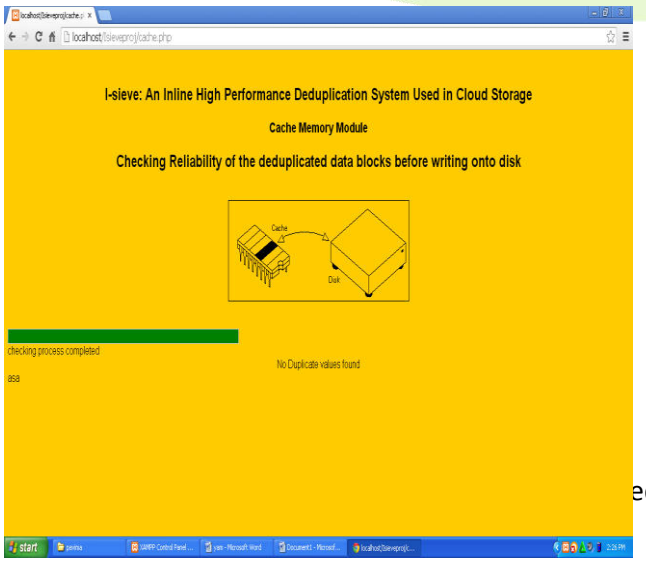

Fig 4 experimental results

VI CONCLUSION

In this project, we propose I-sieve, a high performance inline Deduplication system for use in cloud storage. We design novel index tables to satisfy the I-sieve architecture, since it is a bridge between frontend and backend systems. We also implement a prototype of I-sieve based on iscsi. We present our detailed test results, and demonstrate that I-sieve has excellent foreground performance compared with traditional iscsi applications. In addition, I-sieve is also suitable for Deduplication in small storage environments, especially with virtual machine applications. Finally, I-sieve must co-exist with current de-duplication systems as long as they support the iscsi protocol. In this project, the notion of authorized data de-duplication was proposed to protect the data security by including differential privileges of users. We presented a several new de-duplication constructions supporting authorized de-duplicate check in hybrid cloud architecture, in which the duplicated check tokens of files are generated by the private cloud server with private keys. The security analysis proves that our schemes will be secure in terms of insider and outsider attacks specified in the proposed security model. To protect the genuine data, we must increases the accuracy of data duplicates check, effective hardware capacity utilization by using enhanced techniques.

VII REFERENCES

A. T. Clements, I. Ahmad, M. Vilayannur, and J. Li, Decentralized Deduplication in san cluster file systems, in Proceedings of the 2009 Conference on USENIX Annual Technical Conference, 2009.

B. B. Debnath, S. Sengupta, and J. Li, Chunkstash: Speeding up inline storage Deduplication using flash memory, in Proceedings of the Annual Conference on USENIX Annual Technical Conference, 2010.

C. 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 Proceedings of the 7th Conference on File and Storage Technologies, 2009, pp. 197–210.

D. C. Policroniades and I. Pratt, Alternatives for detecting redundancy in storage systems data, in Proceedings of the Annual Conference on USENIX Annual Technical Conference, 2004.

E. C. Ungureanu, B.Atkin, A.Aranya, S. Gokhale, S. Rago, G. Calkowski, C. Dubnicki, and A. Bohra, Hydrafs: A high-throughput files system for the hydrastor content addressable storage system, in Proceedings of the 8th USENIX Conference on File and Storage Technologies, 2010.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)

International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST)
Vol. 2, Special Issue 15, March 2016

F. D. Bhagwat, K. Eshghi, D. D. E. Long, and M. Lillibridge, Extreme binning: Scalable, parallel Deduplication for chunk-based file backup, in Proceedings of the IEEE International Symposium on Modelling, Analysis Simulation of Computer and Telecommunication Systems, 2009, pp. 1–9.

G. Kruus, C. Ungureanu, and C. Dubnicki, Bimodal content defined chunking for backup streams, in Proceedings of the 8th USENIX Conference on File and Storage Technologies, 2010.

H. Srinivasan, T. Bisson, and G. Goodson, and K. Voruganti, idedup: Latency-aware, inline data Deduplication for primary storage, in Proceedings of the 10th USENIX Conference on File and Storage Technologies, 2012.

I. Dong, F.Douglis, K. Li, H. Patterson, S. Reddy, and P.Shilane, Tradeoffs in scalable data routing for Deduplication clusters, in Proceedings of the 9th USENIX Conference on File and Storage Technologies, 2011.

J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur,Single instance storage in windows 2000, in Proceedings of the 4th Conference on USENIX Windows Systems Symposium, 2000.