Data Backup and Archiving with Enterprise Storage Systems

Slavjan Ivanov[1], Igor Mishkovski[1]

[1]Faculty of Computer Science and Engineering
Ss. Cyril and Methodius University
Skopje, Macedonia

slavjan_ivanov@yahoo.com, igor.mishkovski@finki.ukim.mk

Abstract. Storing data is topic that is constantly present in everyday life, especially in IT industry that is part of all areas today. Today, the need for optimization, data compression and deduplication using enterprise storage systems, has become a necessity. In this paper we describe one of the leading systems for data backup, archiving and deduplication and its algorithm operations (Stream Informed Segment Layout - SISL) used for data backup, archiving and deduplication. We provide a comparison of the best storage solutions manufacturers. Also we present different measurements for backup jobs with different data stream sizes. Additionally we will suggest which data stream size is the best for the deduplication process.

Keywords: SISL, enterprise storage, optimization, compression, deduplication.

1 Introduction

Enterprise storage is a kind of centralized data depository that includes products and services designed to assist large organization with backing up data and archiving. Enterprise storage performs the same functions as smaller data storage solutions, but is more reliable, fault tolerant, and can provide huge amounts of storage. If we have large number of users, heavy workloads, need of high availability and redundancy in our organization, then we definitely should have enterprise storage system [1]. Also enterprise storage systems usually involve centralized storage repositories, such as storage area networks (SANs). These work with fiber channel or network attached storage (NAS) devices. Today, we have many different ways for storing data, such as backup and archiving. IT companies have huge problems with growing data and finding the best storing solution [2]. Also the way of securing data is very important for every type of business. With the size of data growing, the number of corresponding duplicates also grows [4]. A very effective technique for reducing storage costs is the automatic elimination of duplicate data, more commonly referred to as deduplication [3]. Different storage types and solutions use this technique, ranging from archives and backups to random access memory [5]. In [1], authors present modified enterprise storage system and they make different analyses for the system performance. The most popular challenge is identifying and eliminating duplicate data segments on a system that cannot perform fast processing. In [2], authors explain how data deduplication and compression improve the effectiveness of the whole system. Authors in [3], introduce a new solution for backup and archiving data, with similar performances and capabilities like data deduplication process. In [9], Authors describe different techniques for avoiding the disk bottleneck. These are embedded in Data Domain deduplication file system.

Having in mind how the big the data storage concept is (Table 1), and the benefits of using enterprise storage systems with enterprise deduplication algorithms, one of the goals of this work is to describe differences between basic and enterprise deduplication algorithms. Furthermore, we emphasize the benefits of using enterprise storage systems with enterprise deduplication algorithms and present different measurements for backup jobs with different sizes for data stream. In addition, in this work we recommend the adequate data stream size for the deduplication process.

From the following table (Table 1), we can see just how big the data storage market is, in terms of revenue. Note that, the enterprise algorithm described later in this paper is used by the leading company in data storage systems.

This paper is organized as follows. In Section 2 we describe the basic deduplication algorithm and measurements with different data sizes, while in Section 3 we describe the enterprise algorithm for data deduplication. In Section 4 we present the benefits from using storage deduplication and deduplication system with SISL algorithm. Finally, Section 5 concludes this work.

Table 1. Quarterly revenue comparison for top 5 storage vendors

Top 5 Vendors of Worldwide Total Disk Storage Systems, Third Quarter of 2014 (Revenues are in Millions)					
Vendor	***3Q2014 Revenue***	***3Q2014 Market Share***	***3Q2013 Revenue***	***3Q2013 Market Share***	***3Q2014/3Q2013 Revenue Growth***
EMC	$1,822	20,8%	$1,761	21,1%	3,5%
HP	$1,276	14,6%	$1,201	14,4%	6,2%
DELL	$926	10,6%	$865	10,4%	7,1%
IBM	$866	9,9%	$933	11,2%	-7,2%
NetApp	$746	8,5%	$744	8,9%	0,3%
Others	$2,101	24%	$1,993	23,9%	5,4%
All Vendors	$7,738	88.4%	$7,490	90%	5,1%

2 Basic Deduplication Algorithm

Data deduplication, is process that helps when we have data for backup and archiving. The main function of deduplication algorithm is to reduce the storage capacity needed to store amount of data that has to be transferred over a network. These algorithms, practically detect redundancies within a data set or file system and remove them [6]. One of the most important applications of data deduplication are backup and archiving storage systems. These algorithms are able to reduce the storage requirements to a small fraction of the logical backup data size. This work introduces a new extension of the aptly named fingerprinting-based data deduplication [7].

The main idea of the basic algorithm for deduplication is to break the incoming data stream into segments repetitively and compute a unique fingerprint (identifier) for the segment. That identifier is then compared to all others in the system to determine whether it is unique or duplicate. If the identifier is unique, then unique data is stored to disk. If it is not unique, it continues to compare fingerprints. The system gives off the impression of working in its usual way, but in reality it never stores the same segment twice on the disk. This is accomplished by creating new references to the previously stored segment. For maximum data reduction, smaller segments should be used. The smaller the segment, the higher the possibility is for it to be found in more than one place. But smaller segments increase the complexity, because there are more fingerprints for computing and comparing.

The technology that is used in Data Domain systems, uses small segments of data (on average of 8KB), see Fig. 1. This approach results in very good deduplication, which in turn gives us a flexible store, independent from the application [7]. When unique segments are identified, they are compressed with any compression method (LZ, gzip, or others) and are stored to the disk as such.

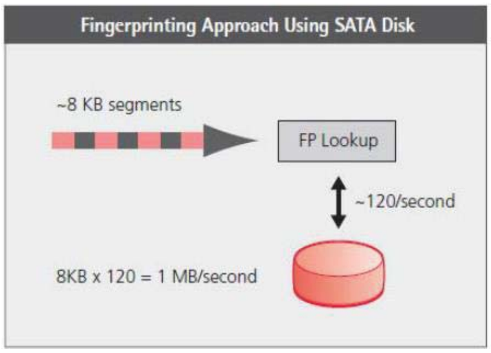


Fig. 1. When using high capacity low cost local SATA disks, the fingerprinting algorithm performing random lookups for segments of average size of 8KB limits the overall throughput to about 1MB/s per disk (Source: EMC Data Domain SISL Scaling Architecture, p. 5)

Because the fingerprint index in this algorithm can be much bigger in size than the amount of the system RAM memory, it is typically stored to disk. For index lookups, the system should perform a disk read for each incoming segment, which may potentially lead to additional problems. This implies that if we want 100 MB/s throughput, we should use hash based system with 100 disks. If SATA disk with size of 500 GB is used, it can support around 120 disk accesses for index lookups per second. If the segment size is 8 KB, a single disk could sustain an incoming data rate of about 120 x 8 KB, or around 1 MB/s. In the next table (Table 2), we took measurements with various data stream sizes. For the purpose of these tests, we used Symantec Backup Exec 2014 backup software, EMC Data Domain DD2500 storage system, interlinked with 10G SAN network. If it is required to go faster, more disks would be required to spread the access load. That kind of system would be too expensive to compete with tape in most situations. In Fig. 2 we present the ratio between different data stream size, average execution time and average execution speed for backup job of 34.5GB.

Table 2. Measurements for backup job of 34.5GB with different data stream sizes

Data Stream Size	Average execution time	Average execution speed
32KB	17min07sec	30.64MB/sec
64KB	15min43sec	38.75MB/sec
128KB	14min10sec	42.80MB/sec
256KB	13min03sec	47.30MB/sec
512KB	12min50sec	48.90MB/sec
1MB	12min16sec	50.05MB/sec

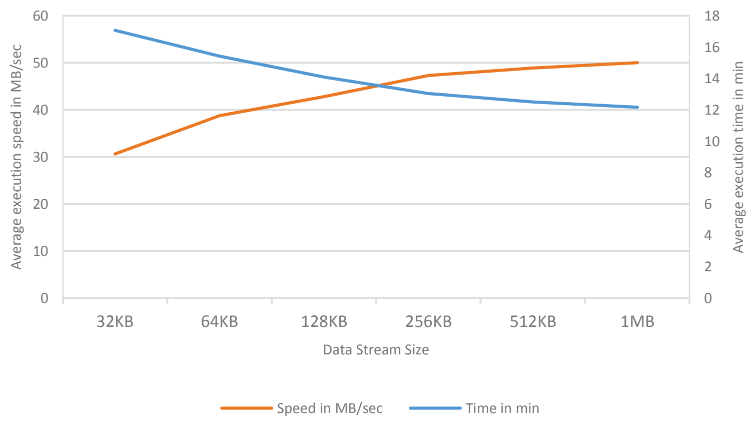


Fig. 2. Graph with different measurements for backup job of 34.5GB

There are more simple solutions. One solution could be to use bigger segment size, but that would provide worse deduplication effects, features and performance. Other solution are the Fibre Channel disks for deduplication, but they often cost 3 to 5 times more than traditional disks.

Thus, if we say that capacity of 100 disks is too much, how many disks are enough?

To determine how fast of a system is needed, the best measurement is to make weekly full backup with daily incrementals. For scalable deduplication system, we need to index the unique identifiers using on-disk structure. For good efficiency, the system needs to effectively seek the disk to determine whether an identifier is new, unique or redundant. At the moment, disk performances require that we have more disks than the required capacity in order to achieve greater seek speeds. The most challenging throughput configuration is when all full backup backups are on one weekend day (Fig. 3).

If we are using a 16-hour weekend backup, at 100 MB/s, a starting dataset would have to be less than 5.75 TB (16h x 100 MB/s). If we have deduplication storage system with 500 GB drives, we only need 12 drives for storage. Also, if we use 1TB disks, then we need half of the previous disks.

The differences in price and manageability between storage system configurations are quite big. There are a lot of differences between systems with 12 disks and 100 disks, especially in capacity and data that is addressed.

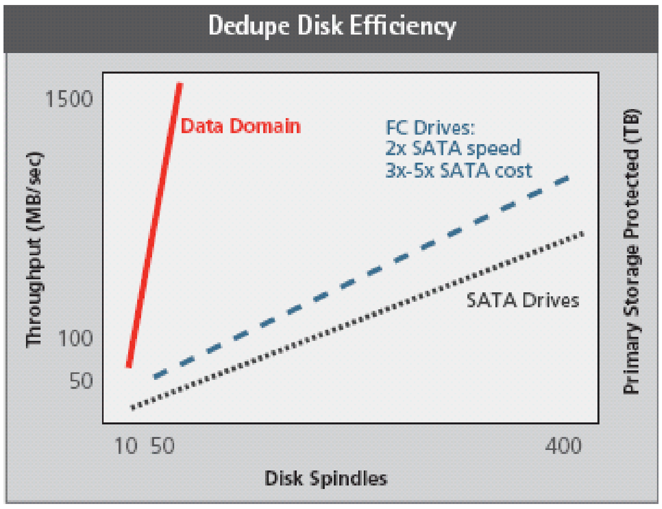


Fig. 3. Deduplication disk efficiency using 500 GB, 7.2k rpm SATA disks and an average 8 KB segment size (Source: EMC Data Domain SISL Scaling Architecture, p. 7)

The disks that are used today, require speeds around 60 MB/s stream. If disks are working with 1 MB/s, the system will need 60 disks for good performance and reading power. To achieve that, the system should minimize RAM usage, enable CPU improvements for improving throughput, avoid disk fragmentation or use enough number of SATA disks to support the required capacity needful for the system.

3 Enterprise Algorithm for Data Deduplication

All previous problems of the deduplication algorithm are solved with the new algorithm that is implemented in EMC Data Domain systems and its name is SISL (Stream Informed Segment Layout). SISL technology includes different combination of deduplication algorithms and it identifies around 99% of the duplicate segments in system RAM memory, before storing them to disk.

SISL makes groups of related segments and fingerprints and store them together in such a way that large groups can be read at once. With these patented techniques, Data Domain enterprise storage system can utilize the full capacity of large SATA disks for data protection and, without increasing RAM, minimize the number of disks required for high throughput. With other words, SISL algorithm allows Data Domain OS - based system performance to track significant CPU speed improvements. SISL implements a series of techniques performed directly in RAM, before storing the data to the disk, for quickly distinguishing between new unique segments and redundant duplicate segments. SISL uses two new terms, summary vector and segment localities [8].

The summary vector is a data object that is part of the memory, used by Data Domain OS for quick identification of unique segments. The process for identification of new segments saves the system from comparing indexes on the disk only to find that the segment is new. The summary vector is array of bits located in system RAM which bits values are initially set to zero. It can effectively determine unique segments without using the fingerprint index. When the system stores a new segment, a few bit values in the array are set to 1. The values are chosen based on the identifier of the segment. When following segment arrives, its bit values are checked. If any of the values are equal to 0, the system will recognize the segment as new, and it can stop looking (Fig. 4).

However, the summary vector is not sufficient for declaring a segment redundant. There is 1% probability, that in one moment all of the segments would have been set to 1 even though the new segment is unique. This happens because of process concurrency. When this happens, the system needs to rely on other mechanisms to conclude recognition, but they are out of the scope of this paper.
The problem with finding duplicates exclusively with index lookups is that every disk access only retrieves one segment. One way to disk efficiency is to retrieve many segments with each access.

Segment localities are units in form of containers. In these, the Data Domain system sequentially stores small segments of data with the same neighboring segments. Most of the time these come together as neighbors before and after the current segment. File system of the Data Domain is a log structured system, at the center of which is located the log of containers that are used for storing localities. Localities are one kind of data segments that are used in backup systems. They are grouped together in containers and appended to their logs [8].

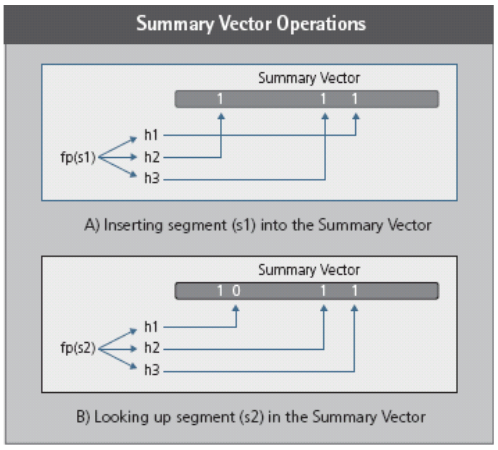

Summary Vector Operations
Summary Vector
1
1
1
h1
fp(s1)
h2
h3
A) Inserting segment (s1) into the Summary Vector
Summary Vector
1 0
1
1
h1
fp(s2)
h2
h3
B) Looking up segment (s2) in the Summary Vector

Fig. 4. The entire array is initialized to 0.When inserting (A), segments with hash identifiers h1, h2, h3 of the segment fingerprint are set to 1. When doing a lookup (B), hash identifiers h1, h2, h3 are checked again and if any are 0, the segment is refused by the system (Source: EMC Data Domain SISL Scaling Architecture, p. 8)

The containers use metadata sections where they store the unique identifiers for the segments, as well as other file system elements. This way the identifiers and the data can be accessed easily when performing deduplication and when the deduplicated stream is reconstructed. The function of locality is keeping segments close together on disk, when they are neighbors. Because of this functionality, the system can access all the identifiers or a whole locality with a single disk access. This means that many associated segments or segment identifiers can be accessed on very effective and fast way (Fig. 5). Upon the arrival of new segment, the system first checks the bits in the summary vector. If the system using the summary vector, finds out that the segment is new, the system adds the segment to the current segment locality for later storage to disk. If it is not a new segment, the system looks in a fingerprint cache in the RAM memory of the system [8].

When using backup and restore processes, most data segments are not often accessed. A full backup passes an entire file system serially through the backup process and references huge numbers of segments that will not be referenced again until the next full backup. For that reason, the technique with caching data that is recently accessed will not be effective.

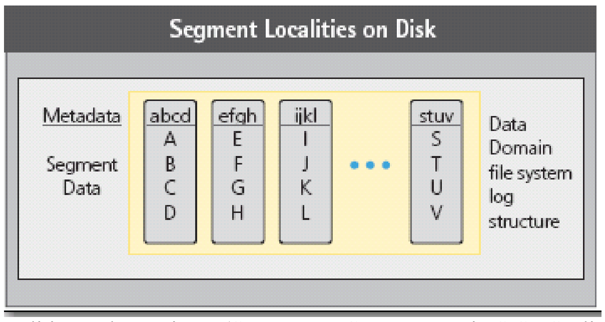
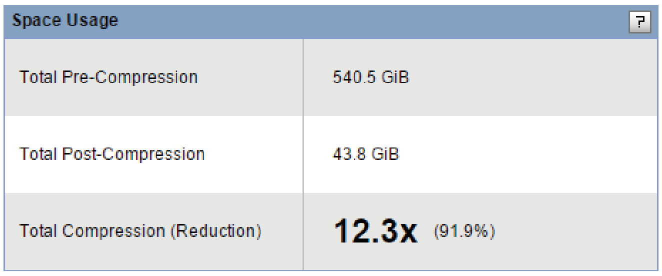

Segment Localities on Disk
Metadata
Segment Data
abcd
A
B
C
D
efgh
E
F
G
H
ijkl
I
J
K
L
stuv
S
T
U
V
Data Domain file system log structure

Fig. 5. Segment localities and containers (Source: EMC Data Domain SISL Scaling Architecture, p. 9)

When using SISL algorithm, if a segment isn't located in the cache, the system looks for it in the on-disk index and then puts the unique identifiers of an entire locality into the cache. Then the following segments in the incoming backup stream are found in the cache of the system and there is no need for further disk accesses. This helps in finding duplicate segments with high speed and reducing hardware needs (including RAM memory and the number of disks). Unnecessary index lookups for new segments are avoided with the use of summary vector. Segments and their unique identifiers are organized as such so that each data fetch consists of only data relevant to that sequence of segments. The preloading of these localities into the cache memory enables fast and effective identifying of duplicate segments. With real backup data, these improvements eliminate up to 98% of the disk reads and provide balanced performance using the full capacity of SATA disks, and getting best deduplication performance [8].

On Fig. 6 we present space usage statistics from our test environment. The tests were made with EMC Data Domain DD2500 storage deduplication system and Symantec Backup Exec 2014 software, both connected with fiber optic in 10G SAN switch. We made a backup of 540.5 GiB data and after deduplication process and the use of SISL algorithm we got 91.9% compression or 12.3 times reduced space.
The total amount of data before compression was 540.5 GiB (Total Pre-Compression). The total data amount of data after compression is 43.8 GiB (Total Post-Compression). This is

perceived as very efficient performance of our storage systems, resulting in more effective usage of storage space and significant cost savings.



Space Usage	
Total Pre-Compression	540.5 GiB
Total Post-Compression	43.8 GiB
Total Compression (Reduction)	12.3x (91.9%)

Fig. 6. Data Domain DD2500 Space Usage statistics

4 Benefits from using storage deduplication system with SISL algorithm

- Reduces the amount of storage space – With high data compression and keeping only unique data, the storage space is reduced and significant capacity saving becomes highly feasible.
- Cost-Effective storage system – The use of low cost disk drives with high performance and CPU scaling architecture make system to be cost effective.
- Fast Backups – Performing deduplication in CPU and RAM and using SISL algorithm, the system can achieve fast backups.
- High deduplication performances – CPU Centric architecture with SISL algorithm and unique segments result with high deduplication performances.
- Less disk accesses – With use of containers and localities, the system reduces disk accesses.
- CPU - Centric Architecture – Uses CPU processing performance for the deduplication process.
- Fewer disk drives – High data compression, reduced amount of storage space and high deduplication performances reduce the number of disk drives in the system.
- Data compression – High compression of data with use of SISL fingerprinting algorithm and fingerprinting unique segments.

5 Conclusion

Today, every company for storage solutions, attempts to reduce cost and get the most out of the system. That is possible with choosing right deduplication system and algorithm. In this paper, we present two algorithms for data deduplication, the first one is the basic algorithm and the second one is optimized algorithm for deduplication and compression, called SISL. The SISL algorithm that is used in Data Domain system architecture optimizes deduplication scalability and performance by minimizing disk accesses. For cost efficiency, the deduplication system needs to use CPU centric architecture and should require small number of disk accesses, so as to minimal number of low-cost, high capacity disks would be utilized. In SISL algorithm, Data Domain system has developed a perfect architecture for providing deduplication storage systems with economical characteristics and hardware with best performance.

Acknowledgement

This work was partially financed by the Faculty of Computer Science and Engineering at the “Ss. Cyril and Methodius”.

References

[1] Yanpei Chen, Kiran Srinivasan, Garth Goodson, Randy Katz, “Design Implications for Enterprise Storage Systems via Multi-Dimensional Trace Analysis”, SOSP 2011-Twenty-Third ACM Symposium on Operating Systems Principles, pp. 43-56.

[2] Steven Scully, Benjamin Woo, “Improving Storage Efficiencies with Data Deduplication and Compression”, Oracle Corporation, 2010, Last accessed: 12 April 2015. < http://www.oracle.com/us/products/servers-storage/7000-family-wp-102825.pdf >.

[3] Tianming Yang, Dan Feng, Jingning Liu, Yaping Wan, “FBBM: A new Backup Method with Data De-duplication Capability”, Multimedia and Ubiquitous Engineering, MUE 2008, International Conference, pp. 30-35.

[4] Dutch T. Meyer, William J. Bolosky, “A study of practical deduplication”, FAST'11 Proceedings of the 9th USENIX conference on File and stroage technologies, 2011, pp. 1-1.

[5] Nagapramod Mandagere, Pin Zhou, Mark A Smith, “Demystifying Data Deduplication”, ACM/IFIP/USENIX Middleware '08 Conference Companion, 2008, pp. 12-17.

[6] Tianming Yang, Hong Jiang, Dan Feng, Zhongying Niu, Ke Zhou, Yaping Wan, “DEBAR: A Scalable High-Performance De-duplication Storage System for Backup and Archiving”, Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium, 2010, pp. 1-12.

[7] EMC Corporation “Deduplication Storage for Backup and Archiving”, 2012.

[8] “EMC Data Domain SISL Scaling Architecture”, EMC Corporation, 2012, Last accessed: 15 April 2015, <http://www.emc.com/collateral/hardware/white-papers/h7221-data-domain-sisl-sclg-arch-wp.pdf >.

[9] Benjamin Zhu, Kai Li, Hugo Patterson, “Avoiding the Disk Bottleneck in the Data Domain Deduplication File System”, Data Domain Inc., FAST'08 Proceedings of the 6th USENIX Conference on File and Storage Technologies, Article No. 18, 2008.