White Paper

EMC Data Domain SISL Scaling Architecture

A Detailed Review

Abstract

While tape has been the dominant storage medium for data protection for decades because of its low cost, it is steadily losing ground to disk-based deduplication storage systems. The CPU-centric design of EMC® Data Domain® systems takes the pressure off of disk I/O as a bottleneck. Over the last 20 years, CPUs have improved in speed by a factor of millions, while disks have improved by about 10x. It appears this performance gap will continue to grow well into the future. It is reasonable to imagine that each doubling of cores could mean Data Domain systems can improve speed by about 50 percent. In Stream-Informed Segment Layout (SISL), Data Domain has developed a proven architecture to deliver high-throughput deduplication storage systems with economical storage hardware. Over time, this will allow the continued scaling of CPUs to add direct benefit to system scalability.

December 2012

EMC²

Copyright © 2012 EMC Corporation. All Rights Reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is.” EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

Part Number h7221.2

Table of Contents

Executive summary 4
CPU-centric, inline deduplication in a compact footprint 4
Introduction 4
Audience 4
The challenge: fingerprint recognition and speed 5
Speed per disk 5
Appropriate capacity 6
Typical results 7
Data Domain SISL 8
Uniqueness identification 8
Redundancy identification and read speed 9
The SISL process 9
Future scalability 10
Conclusion 10

Executive summary

CPU-centric, inline deduplication in a compact footprint

While tape has been the dominant storage medium for data protection for decades because of its low cost, it is losing ground to deduplication in disk systems. Deduplication is an approach that can deliver an order of magnitude greater data reduction than traditional compression over time. This should mean that a deduplication system needs fewer disks. It should also mean that configured costs of a disk storage system are comparable to tape automation.

Customers are sometimes surprised that most emerging deduplication products use a lot more disks than expected. Without careful thought about how to implement it, deduplication can become a disk-intensive activity. The conventional way to increase disk systems' performance is to use more disks and to use faster, more expensive disks. This can spread the load across the relatively low per-spindle access and transfer speeds. Unfortunately, using this approach in a dedupe array can quickly make it more expensive than the tape library against which it will be compared. When using low-cost, high-capacity SATA drives, it would also mean most of the capacity would be wasted, because each disk comes with a lot of space. By adding disks just for better I/O performance, the customer could pay for a lot of unnecessary capacity.

EMC® Data Domain® solved this problem early on with the Stream-Informed Segment Layout (SISL) scaling architecture within the Data Domain Operating System (DD OS). It optimizes deduplication throughput scalability and minimizes disk footprint by minimizing disk accesses. Doing so allows the system throughput to be CPU-centric. Speed increases directly as CPUs improve in performance.

Introduction

This white paper explores the Data Domain SISL approach and its contribution to optimizing deduplication.

Audience

This white paper is intended for EMC customers, technical consultants, partners, and members of the EMC and partner professional services community who are interested in learning more about the Data Domain Stream-Informed Segment Layout (SISL) scaling architecture.

The challenge: fingerprint recognition and speed

Speed per disk

The basic algorithm for deduplication is to break the incoming data stream into segments in a repeatable way and compute a unique fingerprint for the segment. This fingerprint is then compared to all others in the system to determine whether it is unique or redundant. Only unique data is stored to disk. To its clients, the system appears to store the data in the usual way, but internally it does not use disk space to store the same segment repeatedly. Instead, it creates additional references to the previously stored unique segment.

For good data reduction, the segments should be small to maximize the data reduction effect. Smaller segments are more likely to be found in more places. But smaller segments mean there are more segments and therefore more fingerprints to compute and compare. Data Domain deduplication technology uses a relatively small segment size (8 KB average, varying in size). This provides the best deduplication results and provides a flexible, application-independent store. After identifying unique segments, local compression (e.g., LZ, gzip) is applied and only that data is stored to disk.

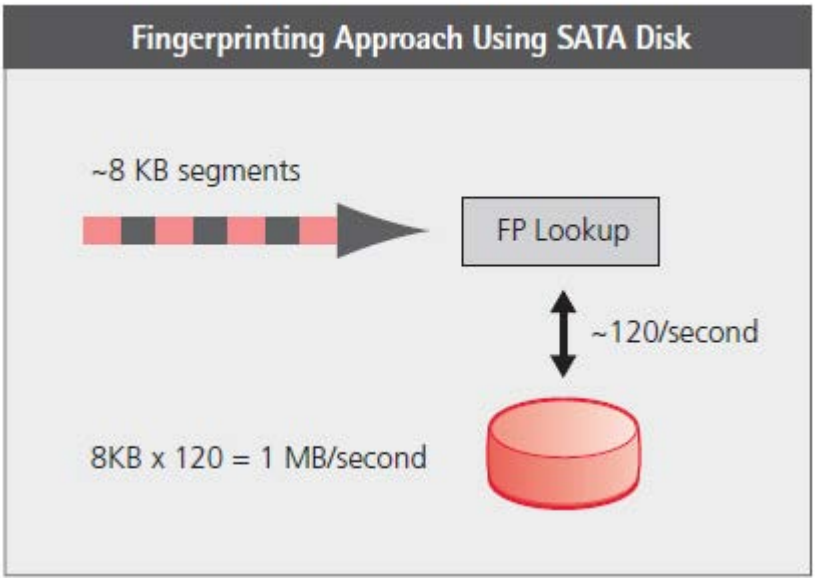

Figure 1. In a fingerprinting approach using high-capacity, low-cost SATA disk, random lookups of fingerprints for segments that average 8 KB limit throughput to about 1MB/s per disk.

The fingerprint index in this kind of approach can be more than an order of magnitude bigger than system RAM. As a result, it is typically stored on disk. So for index lookups, the system will typically perform a disk read for each incoming segment. That is where things can get bad.

This would mean that for 100 MB/s throughput, a typical hash-based system would need about 100 disks. Here's why. A 500 GB SATA disk can sustain approximately

120 disk accesses for index lookups per second[1]. For a segment size of 8 KB, that means a single disk could support an incoming data rate of about 120 x 8 KB, or about 1 MB/s. To go faster, more disks would be required to spread the access load. Such a system would be too expensive to compete with tape in most situations.

Simple alternatives are sub-optimal. One resolution could be to use a much bigger segment size on average, but that would provide significantly worse deduplication effects and again make the system uncompetitive with tape automation on configured price. Or faster Fibre Channel disks could be used – but for twice the speed, they often cost 3x-5x more per gigabyte.

Appropriate capacity

If 100 disks is too much, how much is enough? A dedupe process with traditional compression and a conventional onsite retention period generally achieves ›10x aggregate data reduction. A common onsite retention period stores 10x the amount of data in the starting set (for example, a weekly full backup with daily incrementals, over two months). So it is reasonable to assume the dedupe store should be about as big as the starting set of primary data being backed up to it.

If performance is the limiting factor in matching a dataset to a backup window, the weekly full backup and backup window often determines how fast a system is needed. The most challenging throughput configuration is when all full backups are on one weekend day. If using a 16-hour weekend window to allow for a restart on finding a problem, at 100 MB/s, a starting dataset would have to be less than 5.75 TB (16 hours times 100 MB/s). A dedupe storage system using 500 GB drives should only require 12 drives for storage, apart from RAID, spares, etc. Even with RAID 6, adding two parity disks for a total of 14, this would mean 100 MB/s / 14, or about 7 MB/s / disk. Projecting forward, if each disk stores 1 TB, half as many disks would be needed, so they'd have to go twice as fast to stay on the curve dictated by capacity. Disks themselves will not do this.

[1] For example, a Seagate Barracuda-ES 500 GB drive at 7.2k rpm (model ST3500630NS) supports an average read seek time of 8.5 msec, and so would support 117 read seeks/second.

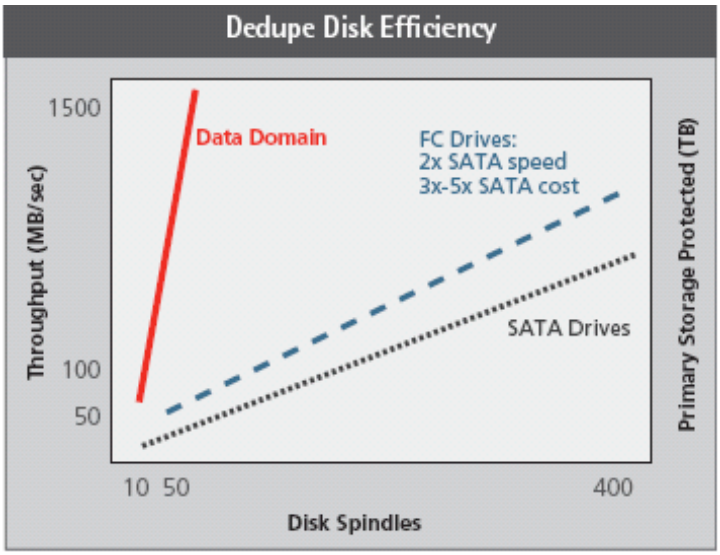

Figure 2. In a scalable deduplication system, fingerprints need to be indexed in an on-disk structure. To achieve speed, the system needs to seek to disk to determine whether a fingerprint is new and unique, or redundant. With current average seek speeds and a small segment size for good compression, more disks are required to get to speed than the desired number for capacity (the figure above assumes 500 GB, 7.2k rpm SATA disks and an average 8 KB segment size).

Typical results

There is obviously a big difference in price and manageability between a configuration that needs 12 drives and one that needs 100 drives. As disks get more capacity and more data is addressed, this difference only becomes more profound. A system that scales speed based on disk drive count will waste enormous amounts of capacity. It would also create a need to manage many more drives than necessary. It will clearly cost more. In addition, if data is spread across drives, accesses would be scattered across all disks and not optimized for read throughput. The store would be fragmented, so the read performance required for recovery or streaming a copy to tape would be compromised.

Modern LTO-4 drives require speeds approaching 60 MB/s / stream or they will shoe-shine and slow down dramatically. At 1 MB/sec / disk, the system would need at least 60 disks (30 TB with 500 GB disks, or 45 TB with 750 GB disks) to perform well enough on reads, not counting additional disks for spares, etc. To succeed, a dedupe system would need to overcome the challenges of comparison across such a large logical space:

- Minimizing RAM
- Using only enough inexpensive SATA disks to support the capacity required by the bandwidth and retention of the system, enabling CPU improvements to directly improve throughput
- Avoiding fragmentation in the disk store to allow fast read, recovery and copy speed

Data Domain SISL

Data Domain SISL technology includes a combination of approaches that solve these problems. First, it identifies 99 percent of the duplicate segments in RAM, inline, before storing to disk. Second, it stores related segments and fingerprints together, so large groups can be read at once. With these patented[2] techniques, Data Domain can utilize the full capacity of large SATA disks for data protection and, without increasing RAM, minimize the number of disks needed to deliver high throughput. In the long term, SISL allows DD OS-based system performance to track dramatic CPU speed improvements.

Uniqueness identification

SISL includes a series of techniques performed inline in RAM, prior to data storage to disk, for quickly filtering both new unique segments and redundant duplicate segments: the *summary vector* and *segment localities*. The summary vector is an in-memory data structure used by DD OS to quickly identify new, unique segments. Identifying new segments saves the system from doing a lookup in the on-disk index only to find the segment is not there. Based on a Bloom filter, the summary vector is a bit array in RAM initially set to all zeros. When a new segment is stored, a few bit locations in the array are set to 1. The locations are chosen based on the fingerprint of the segment. When a subsequent segment arrives, its chosen locations are checked. If any locations are 0, the system knows conclusively that the segment has not previously been stored, and it can stop looking.

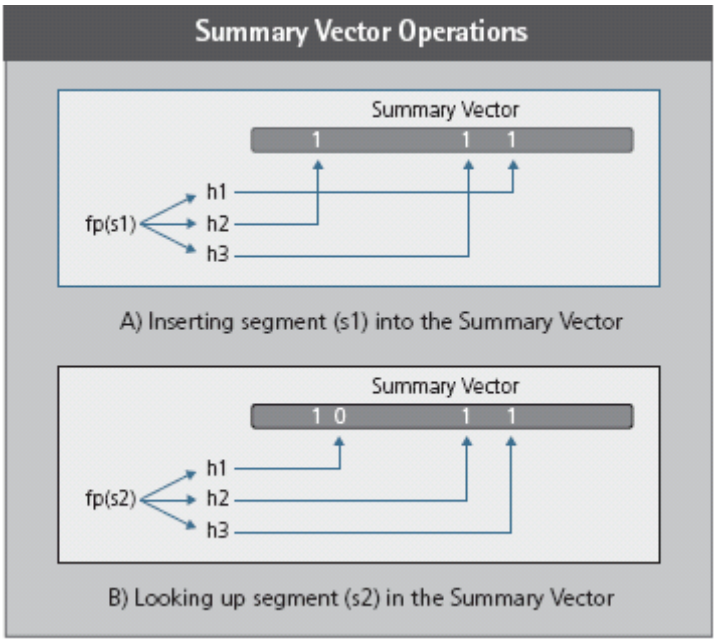

Figure 3. The summary vector can identify most new segments without looking up the segment in the on-disk fingerprint index. Initially all bits in the array are 0. On insertion, shown in (a), bits specified by several hashes, h1, h2, and h3 of the fingerprint of the segment, are set to 1. On lookup, shown in (b), the bits specified by the same hashes are checked. If any are 0, as shown in this case, the segment cannot be in the system.

The summary vector is not, by itself, sufficient for declaring a

[2] These techniques are covered by patents 6,928,526; 7,065,619; and other patents pending.

segment redundant. A small fraction of the time, typically less than 1 percent, all of the chosen locations have been set to 1 by different segments even though the new segment is unique. When this happens, the system needs to rely on other mechanisms to conclude recognition.

Redundancy identification and read speed

The problem with finding duplicates exclusively with index lookups is that every disk access only retrieves one segment. One key to disk efficiency is to retrieve many segments with each access. Generally, a given small segment of data in most backups will tend to be stored sequentially with the same neighboring segments before it and after it most of the time. The Data Domain system stores these neighbors together as sequences of segments in units called *segment localities*, which are packed into containers. The Data Domain file system is a log structured system, at the heart of which is the log of containers storing localities. A locality keeps segments close together on disk when they are neighbors. The system can access all the fingerprints or a whole locality with a single disk access. This means many related segments or segment fingerprints can be accessed very efficiently.

The SISL process

When considering a newly arrived segment, the system first checks the summary vector. If the summary vector indicates the segment is new and needs to be stored, the system, informed by the stream itself, adds the segment to the current segment locality in the order it appears in the stream for later storage to disk. Otherwise, the segment is probably a duplicate and the system looks in a fingerprint cache held in RAM.

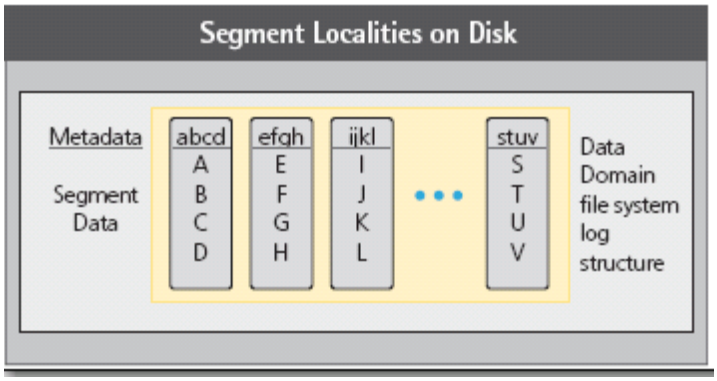

Figure 4. New data segments for a backup stream are stored together in units called localities that, along with their fingerprints and other metadata, are packed into a container and appended to the log of containers. The fingerprints for the segments in the localities are kept together in a metadata section of the container, along with other file system structural elements. This keeps fingerprints and data that were written together close together on disk for efficient access during writes when looking for duplicates and during reads when reconstructing the deduplicated stream.

In backup/restore, most segments are accessed very infrequently. 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. Therefore, a conventional caching strategy based on data recently accessed would not be effective.

With SISL, when a segment is

not found in the cache, the system looks it up in the on-disk index and then prefetches the fingerprints of an entire stream-informed locality into the cache. The vast majority of the following segments in the incoming backup stream are then typically found in the cache without further disk accesses.

Together, these techniques and others make it possible to find duplicate segments at high speed in an application-independent way while minimizing array hardware. It requires neither huge amounts of RAM nor large numbers of disk drives. The summary vector avoids pointless index lookups for new segments. Localities organize segments and segment fingerprints on disk so each disk access fetches data that is relevant for a sequence of segments.

Prefetching brings these localities into the cache so that most duplicate segments are found at high speed in the cache. On long-running experiments with real backup data, these techniques together eliminate up to 98 percent of the disk reads and deliver balanced performance using the full capacity of low-cost SATA disk drives, making inline deduplication possible.

Future scalability

SISL in DD OS takes the pressure off of disk I/O as a bottleneck, so the remaining system design is CPU-centric. That is a good thing. Over the last 20 years, CPUs have improved in performance by a factor of millions, while disks have improved by 10x or so[3]. CPU vendors appear poised to continue these benefits well into the future. It is reasonable to imagine that each doubling of cores could mean Data Domain systems can improve speed by about 50 percent. It is also straightforward to imagine methods to integrate controllers to create multi-node aggregates for larger datasets. A fingerprint-based dialog between controllers can be lightweight. Some approaches exist today that prove the merit in this scalability approach, but they do not solve the price/performance challenges addressed by SISL. Instead, they just make it possible to connect all those spindles needed for good performance. By solving those problems at the outset, the Data Domain architecture provides the foundation for cost-effective aggregation in the future.

Conclusion

Deduplication can help gain an order of magnitude more data reduction than traditional local compression such as LZ or gzip for backup users. But to be cost-neutral when compared to tape automation, the deduplication system needs to be CPU-centric and require very few disk accesses, so that it can be built with the minimum number of low-cost, high-capacity disks. In SISL, Data Domain has developed a proven architecture to deliver high-throughput deduplication storage systems with economical storage hardware. Over time, this will allow the continued scaling of CPUs to add direct benefit to system scalability.

[3] http://seagate.com/docs/pdf/whitepaper/ economies_capacity_spd_tp.pdf