A Survey On Deduplication File System For Virtual Machine Images

Shima V M, Dr. Jayasudha J S

M. Tech Student, SCTCE, pappanamcode, Trivandrum, India HOD, Department of CSE, SCTCE, pappanamcode, Trivandrum, India
Email: vmshimavm@gmail.com

ABSTRACT: Cloud computing is a dominant technology in IT industry for providing different services to users of cloud. One important service is infrastructure as a service. The virtualization is the technique used in infrastructure as a service in which it allocates the virtual machine to user through Internet. Virtual machine which is a guest machine that runs in the host machine environment. Virtual Machine Infrastructure is used to buy Virtual Machine instances to run virtual machines in the cloud platforms. The high overhead of maintaining a virtual machine has been achieved by hardware support such as Intel virtualization technology (VT) by improving the implementation of hypervisor such as KVM, etc. Storage of virtual machine remains a challenging problem because of the high demand of Virtual machine images. A Storage Area Network cannot satisfy the increasing demand of large scale virtual machine hosting for cloud computing because of its cost limitation. The provisioning and depository of a number of Virtual Machine Infrastructure is a challenging problem. The data deduplication is a method used in Virtual Machine Infrastructure and there are various data deduplication techniques which make Virtual Machine Infrastructure storage and provisioning simple and efficient. In this paper, different methods and processes are discussed which is used in deduplication of data to overcome the problems faced in virtual machine images.

Keywords: Virtual Machine, Deduplication, LiveDFS, Liquid.

1 INTRODUCTION

CLOUD computing provides different types of services such as software as a service, platform as a service infrastructure as a service which we can access via Internet. By using virtualization technology the infrastructure as services provides both hardware and software as a service to different users of cloud. The virtual version of server, operating system, hardware, software is created in the virtualization process. Virtual machine looks like a computer which is running within a computer which is known as “guest” machine. The large scale deployment of virtual machine causes the burden in storage and provisioning of the Virtual Machine. This problem can be avoided by using the data deduplication techniques. Virtual Machine Infrastructure formats are supported by different types of hypervisor like VMware, Xen, Virtual box etc. Data deduplication avoids the redundancy of data in storage system which will improve the efficient utilization of storage. In the process of data de-duplication, redundant data is eliminated and only a single instance of the data stored in the database. The most important issue in large scale Virtual Machine Infrastructure deployment and provisioning is storage consumption where existing system solve the issue by the storage area network cluster deduplication. The storage area network is very expensive and does not satisfy the large scale deployment of Virtual Machine Infrastructure. The Virtual Machine provisioning is done with structuring and deploying of the Virtual Machine Images before the hosting of Virtual Machine. During the hosting of Virtual Machine the storage is not efficient due to the duplication of data. Data duplication means there are number of duplicate copies of same data in the datacenter which leads to lower efficiency. In order to overcome this problem the data deduplication is used. This is a specialized data compression technique which is used for eliminating duplicate copies of data.The technique of data deduplication, which is used for storing single instance of duplicate data, eliminates the redundant data in the storage. It is used to decrease the size of data center and reduce the replications of data that were duplicated on cloud. The data de-duplication process which helps to eliminate any block that are not unique and store in smaller group of block

2 Familiarizing Terms

2.1 Virtual Machine

Multiple operating system instances run on a single computer system by using Virtual machine mechanism. Xen, VMware, and QEMU systems provide an execution environment to “guest” operating systems. It uses many techniques to convince each guest operating system that it has control of a full computer system. Virtual Machine uses the technique that uses a file in the “host” operating system to store the contents of the guest’s disk[6]. The size of virtual disk images is specified when the virtual machine is created. That is flat or sparse images. Flat images include fixed-size files in which one block for each block in the guest’s disk. The blocks are zero-filled initially. Thus there are a lot of zero-filled blocks in flat disk images. Sparse images are different from flat images. Sparse images contains only virtual disk blocks that have been written at least once. Therefore sparse images can take little space compared to flat images when created. When the guest operating system uses more and more of its virtual disk the space grows accordingly. That is sparse images can contain zero-filled blocks. When the guest operating system may write a block containing all zeros; such a block would be stored by the virtual machine in the disk image file. The specifications for one file Format is available from VMware while there is no global standard for virtual machine disk images.

2.2 Data deduplication

In data deduplication the objects usually files or blocks are compared and remove all duplicate copies of objects which are non-unique[3]. In the deduplication process, smaller group of blocks being stored by removing non-unique contents. The files are converted into small segments. Then the redundancy is checked for new and existing data. Meta data are updated, data integrity is checked, segments are compressed and then duplicate data are deleted. The files are broken down to segments. This can be done in two ways: fixed size chunking and variable size chunking. In fixed size chunking original files are split into blocks in same size. The method used in variable size chunking is Rabin fingerprint on file content. It also detects the boundaries inside the file. In deduplication most

probably fixed size chunking is used.

A) Data deduplication process

1) Offline data deduplication: The deduplication process[15] is performed after storing the data in storage disk or data center.
2) Online data de-duplication: In Online data deduplication the dedepulication process is performed before storing the data in storage disk or datacenter.

Data deduplication are of different types .They are explained a follows

- Target-based deduplication: Target based deduplication which is used to eliminate the duplicate copies from data after transmission to backup. It will reduce only the storage space and will not reduce the size of data. By using target based deduplication storage space is reduced and it does not save the bandwidth.
- Source-based deduplication: In Source based deduplication the deduplication is done before the transmission of data to target backup. Source deduplication will increase both bandwidth and storage efficiency. Source deduplication will store only the unique data and it will never sent the duplicate data through network. It is used only for reducing the duplicate content of large volume of data.

B) Data deduplication levels

1) **File-level deduplication:** The deduplication is performed on a single file. The files are checked based on their hash values. In this method the duplicate files are identified. The hash numbers are relatively easy to generate. So it requires less processing power.

2) **Block-level deduplication:** The deduplication is performed over blocks in Block level deduplication process. The files are first divided into blocks and then store only a single copy of each block. This type may use fixed size chunking or variable size chunking. When compared to whole file, the block level deduplication eliminates the small redundant chunk of data. The same deduplication algorithm is used by each and every file system in block level deduplication.

3) **Byte-level deduplication**: Byte-level deduplication understands the content, or "semantics", of the data which is a form of block-level deduplication. Byte-level deduplication efficiently deduplicates the bytes within the data stream and it understands the content of the data. Factors that affect deduplication are Primary Storage Deduplication, File types, Virtualized Environments etc.

3 DEDUPLICATION FILE SYSTEMS

The deduplication storage systems are of several types which are designed for eliminating duplicate copies in storage systems. The similarity between them is based on storage system.

3.1 HydraFS

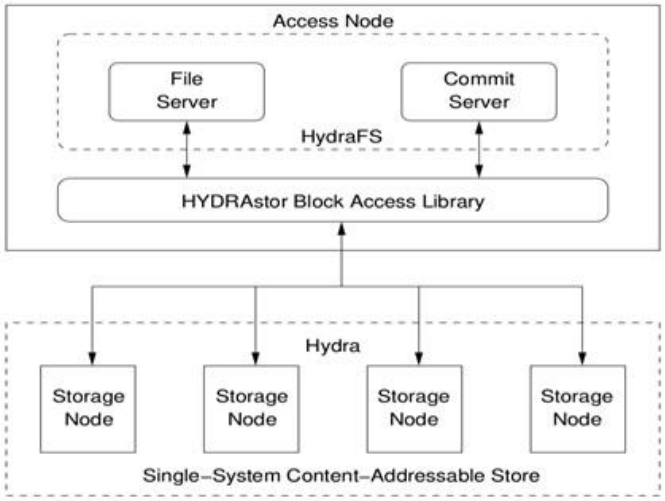

Fig 1: *HydraFS Architecture*

HydraFS is a file system built on top of HYDRAstor which is scalable, distributed, content addressable block storage system [2]. HydraFS provides high performance for reads and writes. Performance is 82–100% for streaming access of the HYDRAstor throughput. It performs high duplicate elimination by using deduplication process. Hydra has multiple numbers of nodes. The storage system is content addressable[7]. It stores blocks at configurable redundancy levels For streams of large blocks of read and write it provides high throughput. Hydra is designed to provide a content-addressable block device interface that hides the details of data distribution and organization. It has also the facility of removal of nodes and handling of disk and node failures. HydraFS supports high-bandwidth for streaming workloads. Its first commercial application is as part of a backup appliance. HydraFS acts as a front end for the Hydra distributed content-addressable block store as shown in figure 1[2].

3.2 LiveDFS

Live Deduplication File System is for deduplication storage of Virtual Machine images in an open source cloud[1]. LiveDFS is designed based on commodity hardware and operating system. LiveDFS is implemented under low cost. LiveDFS can save up to 40% of storage space for VM images. The implementation of LiveDFS in cloud is based on open stack. It is based on inline deduplication. It is applicable to single storage partition. The important features of LiveDFS are spatial locality, prefetching of metadata and journaling. LiveDFS is implemented as a Linux kernel-space file system.

LiveDFS Design: LiveDFS is a file system that implements inline deduplication for virtual machine storage. It is implemented as a storage layer for an open-source cloud. It is generally designed for commodity hardware and operating systems. LiveDFS is implemented in a single storage partition. Deduplication is applied to the data stored within the same partition but not for the same data stored in different partitions. It is possible for a partition to have multiple storage devices. At file-system level the deduplication is applied, while data chunking is applied in the storage devices and is transparent to the file system. LiveDFS aim at virtual machine image storage.

File system layout: The general file system I/O operations

such as read, write and delete are supported in LiveDFS[1]. It also supports other standard file system operations for files, directories and metadata. Figure 1 depicts the layout of file system LiveDFS that allows block sharing. The storage systems organize data into different blocks which could be of fixed size or variable size chunks. The merits of using variable-size blocks in backup systems are given in [7]. In LiveDFS blocks are arranged into block groups (as given in Figure 2)[1]. The metadata of the blocks are stored within the same group. The block groups are used to reduce the distance between the metadata and their corresponding blocks on disk thereby saving the disk seek overhead. An inode holds a set of block pointers, which store the block numbers (or addresses) of the blocks associated with the file.

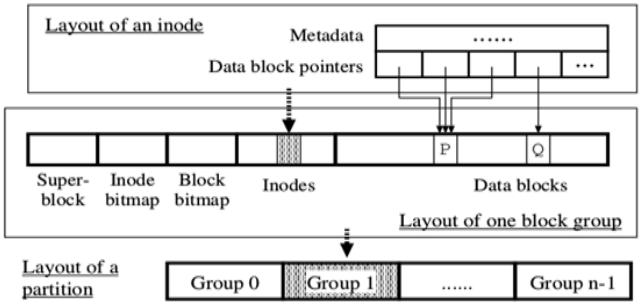

Fig 2: *LiveDFS File System Layout*

3.3 Liquid

Liquid is a distributed file system that is used for deployment of Virtual Machines. Its client side breaks Virtual Machine images into small data blocks, references them by their fingerprints (calculated during a deduplication process), and uses deduplication techniques to avoid storing redundant data blocks[4]. There are mainly three components for Liquid[4][16]. They are single meta server with hot backup, multiple data servers and multiple clients. These machines run a user level service process in a Linux machine. The Virtual Machine (VM) images are divided into data blocks. The data blocks have fixed size. Every data blocks has its own irreplaceable fingerprint. This fingerprint is calculated at same point in deduplication process. Generally, the Liquid represents a Virtual Machine image using successive fingerprints that points to data blocks. The meta server contains information about the file system layout. To maintain the availability, the server takes backup to a shadow meta server. The data server monitors data blocks. These data stored in a distributed hash table (DHT) [3]. Every data server has a range of fingerprint given by meta server. The meta server checks the availability of data server. After all process, the Virtual Machines are shutting down; the client uploads changed metadata to meta server. The Liquid gives the option for fault tolerance. But when meta server crashes the backup server will takeover its place. Duplicates of data blocks are stored in data servers, crashing some of data servers will not disturb the whole system.

File system layout:

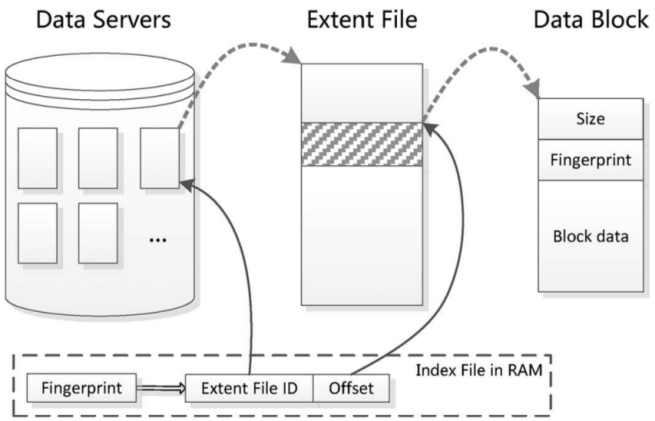

Fig 3: Liquid File System LayOut

The meta data of all file system are stored on a meta server. The meta data of each virtual machine image are stored in the meta server[4]. The number of data blocks and corresponding meta data are stored in order to reduce tracking overhead. There are a lot of protocols available for virtual machine images. But most of these existing protocols do not utilize deduplication for virtual machine images. Each data server is a valid data block provider. It publishes a Bloom filter. In a Bloom filter the fingerprints of all their data blocks are stored. The Bloom filter uses an array of bits. Initially all the bits are set to 0 to represent the existence information of fingerprints. Different hash functions are used. Each hash function maps a fingerprint to one of the array positions randomly with a uniform distribution. While adding a new fingerprint into the Bloom filter the hash functions are used to map the new fingerprint into bits in the bit vector which will be set as 1. It is done so as to indicate its existence. To search for a fingerprint the hash functions are applied to get the array positions.

4 Conclusion

In an environment with large numbers of VM disk images, Deduplication is an efficient approach since it reduces storage demands. The deduplication of Virtual Machine disk images can save more of the space required to store the operating system and application environment. The deduplication file system with good IO performance has a rich set of features. Peer to peer technique can be used to improve sharing of data blocks that makes the system highly scalable. In drives, spinning deduplication can be used in order to reduce the amount of data storage. Deduplication on Virtual Machine images is proved to be highly effective and efficient.

References

[1] J.S. Bridle, “Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition,” Neurocomputing—Algorithms, Architectures and Applications, F. Fogelman-Soulie and J. Herault, eds., NATO ASI Series F68, Berlin: Springer-Verlag, pp. 227-236, 1989. (Book style with paper title and editor)

[2] W.-K. Chen, Linear Networks and Systems. Belmont, Calif.: Wadsworth, pp. 123-135, 1993. (Book style)

[3] H. Poor, “A Hypertext History of Multiuser Dimen-

sions," MUD History, http://www.ccs.neu.edu/home/pb/mud-history.html. 1986. (URL link *include year)

[4] K. Elissa, “An Overview of Decision Theory," unpublished. (Unplublished manuscript)

[5] R. Nicole, "The Last Word on Decision Theory," J. Computer Vision, submitted for publication. (Pending publication)

[6] C. J. Kaufman, Rocky Mountain Research Laboratories, Boulder, Colo., personal communication, 1992. (Personal communication)

[7] D.S. Coming and O.G. Staadt, "Velocity-Aligned Discrete Oriented Polytopes for Dynamic Collision Detection," IEEE Trans. Visualization and Computer Graphics, vol. 14, no. 1, pp. 1-12, Jan/Feb 2008, doi:10.1109/TVCG.2007.70405. (IEEE Transactions)

[8] S.P. Bingulac, “On the Compatibility of Adaptive Controllers,” Proc. Fourth Ann. Allerton Conf. Circuits and Systems Theory, pp. 8-16, 1994. (Conference proceedings)

[9] H. Goto, Y. Hasegawa, and M. Tanaka, “Efficient Scheduling Focusing on the Duality of MPL Representation,” Proc. IEEE Symp. Computational Intelligence in Scheduling (SCIS ’07), pp. 57-64, Apr. 2007, doi:10.1109/SCIS.2007.367670. (Conference proceedings)

[10] J. Williams, “Narrow-Band Analyzer,” PhD dissertation, Dept. of Electrical Eng., Harvard Univ., Cambridge, Mass., 1993. (Thesis or dissertation)

[11] E.E. Reber, R.L. Michell, and C.J. Carter, “Oxygen Absorption in the Earth’s Atmosphere,” Technical Report TR-0200 (420-46)-3, Aerospace Corp., Los Angeles, Calif., Nov. 1988. (Technical report with report number)

[12] L. Hubert and P. Arabie, “Comparing Partitions,” J. Classification, vol. 2, no. 4, pp. 193-218, Apr. 1985. (Journal or magazine citation)

[13] R.J. Vidmar, “On the Use of Atmospheric Plasmas as Electromagnetic Reflectors,” IEEE Trans. Plasma Science, vol. 21, no. 3, pp. 876-880, available at http://www.halcyon.com/pub/journals/21ps03-vidmar, Aug. 1992. (URL for Transaction, journal, or magzine)

[14] J.M.P. Martinez, R.B. Llavori, M.J.A. Cabo, and T.B. Pedersen, "Integrating Data Warehouses with Web Data: A Survey," IEEE Trans. Knowledge and Data Eng., preprint, 21 Dec. 2007, doi:10.1109/TKDE.2007.190746.(PrePrint)

[15] Offline Deduplication for Btrfs. http://www.spinics.net/lists/ linux-btrfs/msg07818.html.

[16] A.T. Clements, I. Ahmad, M. Vilayannur, and J. Li, ‘‘Decentralized Deduplication in San Cluster File Systems,’’ in Proc. Conf. USENIX Annu. Techn. Conf., 2009, p. 8, USENIX Association.