Volume 4, Issue 10, October 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

An Efficient Storage Consumption of Virtual Machine Images by Deduplication File System

M. Sindhuja*
PG C.S.E
Arasu Engineering College
Kumbakonam, India

R. Rajakumar
Dept of C.S.E
Annai Group of Institutions
Kumbakonam, India

M. Kannan
Dept of C.S.E
Arasu Engineering College
Kumbakonam, India

Abstract— Cloud computing is an evolving technology which enables sharing of resources on large scale. The role of virtual machine creates betterment in cloud environment because of its rich set of features. One of the major challenges in cloud computing is due to increasing demand of virtual machine Image storage. Existing system strives to reduce the storage consumed by virtual machine images by Decentralized Deduplication in storage area network cannot fulfill the storage requirement in future. In this paper, we put forward a deduplication file system with peer-to-peer data transfer. Deduplication in this system is carried out by storage of fingerprints of data blocks based on Bloom filter. It comprises certain efficient features related to storage of VM images by local caching, on-demand fetching.

Keywords— Cloud computing, deduplication, file system, liquid, peer to peer, storage, virtual machine

I. INTRODUCTION

Cloud computing is internet-based computing in which large groups of remote servers are networked to allow the centralized data storage Cloud computing relies on sharing of resources to achieve coherence and economies of scale, similar to a utility (like the electricity grid) over a network.[1] At the foundation of cloud computing is the broader concept of converged infrastructure and shared services. Cloud computing, or in simpler shorthand just "the cloud", also focuses on maximizing the effectiveness of the shared resources.

Cloud resources are usually not only shared by multiple users but are also dynamically reallocated per demand. This can work for allocating resources to users. For example, a cloud computer facility that serves European users during European business hours with a specific application (e.g., email) may reallocate the same resources to serve North American users during North America's business hours with a different application (e.g., a web server). This approach should maximize the use of computing power thus reducing environmental damage as well since less power, air conditioning, rack space, etc. are required for a variety of functions. With cloud computing, multiple users can access a single server to retrieve and update their data without purchasing licenses for different applications.

Proponents claim that cloud computing allows companies to avoid upfront infrastructure costs, and focus on projects that differentiate their businesses instead of on infrastructure. Proponents also claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and enables IT to more rapidly adjust resources to meet fluctuating and unpredictable business demand.[2][3][4] Cloud providers typically use a "pay as you go" model. This can lead to unexpectedly high charges if administrators do not adapt to the cloud pricing model. The present availability of high-capacity networks, low-cost computers and storage devices as well as the widespread adoption of hardware virtualization, service-oriented architecture, and autonomic and utility computing have led to a growth in cloud computing. Cloud vendors are experiencing growth rates of 50% per annum.In cloud computing environment better management of resources is provided by means of virtual machines.The purpose of VM is to improve resource sharing among users and to enhance the performance in terms of application flexibility. This virtualization technology has been revitalized as the need for cloud computing in recent year.

New virtual machines are created based on virtual machine images. Due to increasing number of virtual machines being deployed, it creates load on existing storage system. Existing systems have made efforts to address this storage problem by deduplication. Deduplication is a technology that reduces storage consumption by eradicating duplicate copies of data and it stores only unique data blocks. Such data blocks are determined with the help of fingerprints. The fingerprint is a cryptographic hash value calculated by SHA-1 plays a massive role in identifying duplicate data blocks. Data block is considered as repeated if there is an occurrence of same fingerprint. Instead of storing the same content for many times, only the reference of such repeated data block is stored. This technique could be more effective when there is a redundant dataset.

A. Virtual Machine

In computing,a virtual machine (VM) is an emulation of a particular computer system. Virtual machines operate based on the computer architecture and functions of a real or hypothetical computer, and their implementations may involve specialized hardware, software, or a combination of both. Classification of virtual machines can be based on the degree to which they implement functionality of targeted real machines. That way, system virtual machines (also known

as full virtualization VMs) provide a complete substitute for the targeted real machine and a level of functionality required for the execution of a complete operating system. On the other hand, process virtual machines are designed to execute a single computer program by providing an abstracted and platform-independent program execution environment.

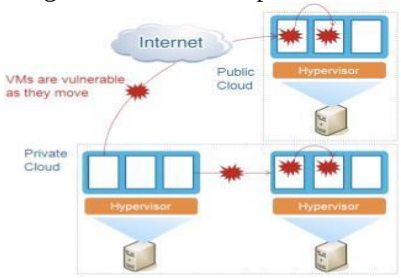

Fig. 1 Cloud Computing

II. BACKGROUNDS

A. VM Image Formats

There is two basic formats for VM images. The raw image format is simply a byte-by-byte copying of physical disk's content into a regular file(or a set of regular files). The benefit of raw format images is that they have better IO performance because their byte-by-byte mapping is straightforward. However, raw format images are generally very large in size, since they contain all contents in the physical disks, even including those blocks which never really get used. The other one is the sparse image format. Instead of simply doing a byte-by-byte copy, the sparse image format constructs a complex mapping between blocks in physical disks and data blocks in VM images. For special blocks, such as those containing only zero bytes, a special mark is added on the block mapping, so that the blocks do not need to be stored, since their content could be easily regenerated when necessary. This will help reduce the size of newly created VM images, since most blocks inside the images would never be used, which only contain zero bytes, and hence, do not need to be stored. Manipulations on the block mapping and a VM image bring advanced features such as snap shotting, copy-on-write images [2], etc. However, the block mapping in sparse format also results in worse performance of IO compared with raw images. Interpreting the block mapping introduces additional overhead, and it generally breaks sequential IO issued by hypervisors into random IO on VM images, which significantly impairs IO performance. Both formats are widely used in hypervisors such as Xen, KVM, Virtual Box, etc. All these hypervisors support raw format natively. Xen and KVM support the qcow2 sparse format, which has two levels of block mapping and a block size of 256 KB. Virtual Box supports the vdi sparse format, which has one level of block mapping and a coarser granularity of data block size at 1 MB. It aims at improving storage utilization. In the process of deduplication, unique blocks of data are usually identified by an analyzed fingerprint from their content.

Whenever the fingerprint of a data block is calculated, it is compared with a stored fingerprint database to check for a math. This data block will be defined as a redundant data block, if an identical fingerprint is found. A redundant data block is replaced with a reference to the stored data block, instead of storing the same content multiple times. This technique will be highly effective when the original data set is redundant. For archival systems, research has shown that deduplication could be more effective than conventional compression tools [5]. The basic unit for deduplication could be a whole file, or sections inside a file. For the latter case, there are two methods to break a file into sections, namely, fixed size chunking and variable size chunking [14]. The fixed size chunking method splits the original file into blocks of the same size (except the last block). The variable size chunking method adopts a more complicated scheme, by calculating Rabin fingerprint [5] of a sliding window on file content, and detects more natural boundaries inside the file.

Compared with variable size chunking, fixed size chunking will have better read performance, but it cannot handle non-aligned insertion in files effectively. Variable size chunking has been widely applied in archival systems, which deals with data that are rarely accessed [3]. For VM images, research has shown that fixed size chunking is good enough in measure of deduplication ratio [7], [2].

III. METHODOLOGIES

Bloom filter is used to store the fingerprints of data blocks compactly. It makes use of hash functions to map those fingerprints to one of the array positions. To insert a new fingerprint it makes use of hash functions for mapping and results out its existence. It also rely on hash function to query a fingerprint. Here comes the storage of fingerprint is handled by multi dimensional bloom filter.

A. System model

The system architecture contains main components like Meta server, Data server (DS), clients.VM images are divided in to blocks of fixed sizes. Then the fingerprints for data blocks are calculated .The Meta server is responsible for maintaining fingerprint, mapping of fingerprint to data server. The information present in the Meta server is replicated in shadow Meta server to ensure fault tolerance and availability. If the Meta server crashes, shadow meta server will perform entire functionality of the meta server. Data server (DS) is responsible for managing data blocks of VM images. Client side of the file system maintains cache for storing recently accessed Meta data of data blocks. It leads to faster access of data. During deduplication technique, data blocks are referenced by fingerprints to avoid storing the redundant data. The deduplicated data blocks are stored in group of data server.

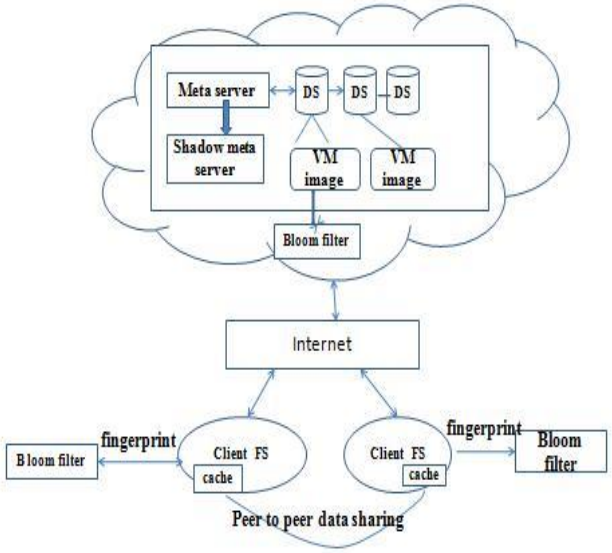

Fig. 2 System Architecture

IV. DESIGNS AND IMPLEMENTATION

A. Assumptions

Here are a few assumptions for the VM images and their usage when Liquid is designed.

1. A disk image will only be attached to at most one Running VM at any given time.
2. The disk images are mainly used to store OS and Application files. User generated data could be stored directly into the disk image, but it is suggested that large pieces of user data should be stored into other storage systems, such as SAN. Temporary data could be saved into ephemeral disk images on DAS.

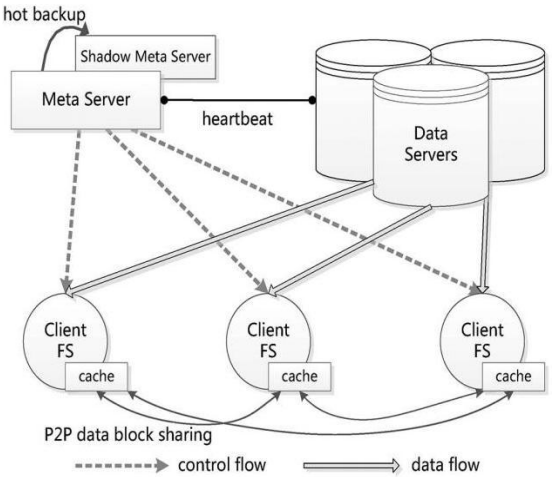

Fig. 3 Liquid Architecture

B. Data Deduplication

In computing**,** data deduplication is a specialized data compression technique for eliminating duplicate copies of repeating data. Related and somewhat synonymous terms are intelligent (data) compression and single-instance (data) storage. This technique is used to improve storage utilization and identify large sections – such as entire files or large sections of files – that are identical, in order to store only one copy of it. This copy may be additionally compressed by single-file compression techniques. For example a typical email system might contain 100 instances of the same 1 MB (megabyte) file attachment. Each time the email platform is backed up, all 100 instances of the attachment are saved, requiring 100 MB storage space. With data deduplication, only one instance of the attachment is actually stored; the subsequent instances are referenced back to the saved copy for deduplication ratio of roughly 100 to 1.

C. Deduplication in Liquid

1) Fixed sized chunking: Liquid chooses fixed size chunking instead of variable size chunking. This decision is made based on the observation that most x86 OS use *a* block size of 4 KB for file systems on hard disks. Fixed size chunking applies well to this situation since all files stored in VM images will be aligned on disk block boundaries. Moreover, since OS and software application data are mostly read-only, they will not be modified once written into a VM image. The main advantage of fixed size chunking is its simplicity. Storing data blocks would be easy if they have the same size, because mapping from file offset to data block could be done with simple calculations. Previous study [12] has shown that fixed size chunking for VM images performs well in measure of deduplication ratio.

2) Optimization on Fingerprint Calculation: Deduplication systems usually rely on comparison of data block fingerprints to check for redundancy. The fingerprint is a collision-resistant hash value calculated from data block contents. MD5 [26] and SHA-1 [12] are two cryptography hash functions frequently used for this purpose. The probability of fingerprint collision is extremely small, many orders of magnitude smaller than hardware error rates [2]. So we could safely assume that two data blocks are identical when they have the same fingerprint.

3) Storage for Datablocks: Deduplication based on fixed size chunking leads to numerous data blocks to be stored. One solution is to store them directly into a local file system, as individual files. This approach will lead to additional management overhead in a local file system. Even though file systems such as ReiserFS [6] and XFS [7] have been designed to be friendly to small files, there will still be overhead on the frequent open(), close() syscalls and Linux kernel vnode layer. Moreover, most Linux file system implementations use linked ists to store meta data of files under a directory [12], so file look-up will have a time complexity.

4) Block Size Choice: Block size is a balancing factor which is very hard to Choose, since it has great impact on both deduplication ratio and IO performance. Choosing a smaller block size will lead to higher deduplication ratio, because modifications on the VM images will result in smaller amount of additional data to be stored. On the other hand, smaller block size leads to more data blocks to be analyzed. When block size is too small, the sheer number of data blocks will incur significant management overhead, which impairs IO performance greatly. Moreover, smaller block size will result in more random seeks when accessing a VM image, which is also not tolerable. Choosing block size smaller than 4 KB makes little sense because most OS align files on 4 KB boundaries. A smaller block size will not be likely to achieve higher deduplication ratio. A large block size is not preferable either, since it will reduce deduplication ratio, although the IO performance will be much better than a small block size. Liquid is compiled with block size as a parameter. This makes it more adaptive to choose different block size under different situation. Based on our experience, it is advised to use a multiplication of 4 KB between 256 KB and 1 MB to achieve good balance between IO performance and deduplication ratio.

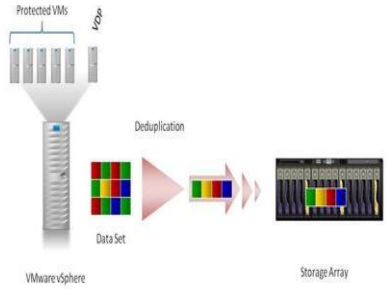

Fig. 4 Data Deduplication

D. Post-process deduplication

With post-process deduplication, new data is first stored on the storage device and then a process at a later time will analyze the data looking for duplication. The benefit is that there is no need to wait for the hash calculations and lookup to be completed before storing the data thereby ensuring that store performance is not degraded. Implementations offering policy-based operation can give users the ability to defer optimization on "active" files, or to process files based on type and location. One potential drawback is that you may unnecessarily store duplicate data for a short time which is an issue if the storage system is near full capacity.

E. Post-process deduplication

This is the process where the deduplication hash calculations are created on the target device as the data enters the device in real time. If the device spots a block that it already stored on the system it does not store the new block, just references to the existing block. The benefit of in-line deduplication over post-process deduplication is that it requires less storage as data is not duplicated. On the negative side, it is frequently argued that because hash calculations and lookups takes so long, it can mean that the data ingestion can be slower thereby reducing the backup throughput of the device.

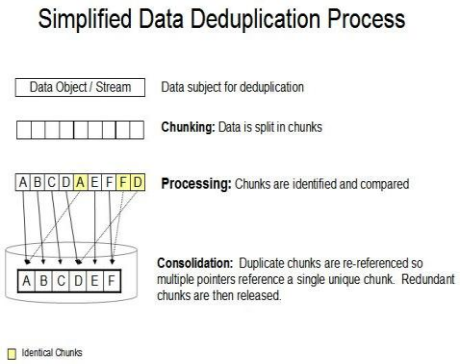

Fig. 5. Simplified Data Deduplication

V. FILE SYSTEM LAYOUTS

All file system Meta data are stored on the Meta server. Each VM image's Meta data are stored in the Meta server's local file system as individual files, and organized in a conventional file system tree. User-level applications are provided to fetch VM Meta data files stored on the meta server, or to publish a newly created VM image by pushing its meta data to the meta server.

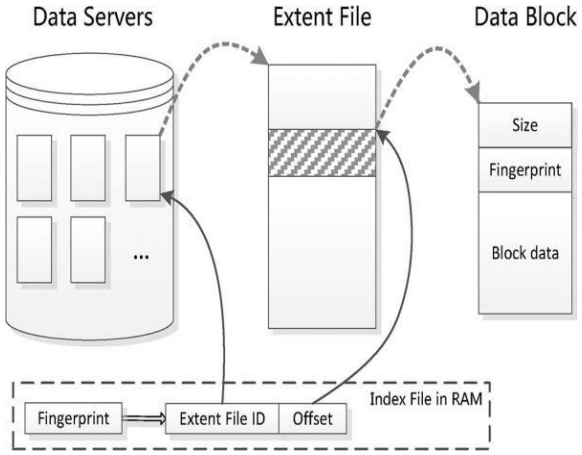

Fig. 6 Process of look-up by fingerprint.

The statistic results are obtained by reading data blocks stored in client side of Liquid file system's local cache. A smaller data block size results in more frequent random access operations, and in turn degrades IO performance. With the increase of data block size, IO performance gradually improves, and stabilizes after it reaches 256 KB.

VI. CONCLUSIONS

We have presented Liquid, which is a deduplication file system with good IO performance and a rich set of features. Liquid provides good IO performance while doing deduplication work in the meantime. This is achieved by caching frequently accessed data blocks in memory cache, and only run deduplication algorithms when it is necessary. By organizing data blocks into large lumps, Liquid avoids additional disk operations incurred by local file system. Liquid supports instant VM image cloning by copy-on write technique, and provides on-demand fetching through Network, which enables fast VM deployment. P2P technique is used to accelerate sharing of data blocks, and makes the system highly scalable. Periodically exchanged Bloom filter of data block fingerprints enables accurate tracking with little network bandwidth consumption. Deduplication on VM images is proved to be highly effective. However, special care should be taken to achieve high IO performance. For VM images, parts of an image are frequently modified, because the OS and applications in VM are generating temporary files. Caching such blocks will avoid running expensive deduplication algorithms frequently, thus improves IO performance. Making the system scalable by means of P2P technique is challenging because of the sheer number of data blocks to be tracked. By compacting data block fingerprints into Bloom filters, the management overhead and Meta data transferred over network could be greatly reduced.

VII. FUTURE WORK

Traditionally bloom filter is implemented by single array of m bits. But multi dimensional bloom filter make use of multi dimensional bit vector to improve the data storage and avoids the duplication in large file system effectively.

REFERENCES

[1] Amazon Machine Image, Sept. 2001. [Online]. Available: http://en. wikipedia.org/wiki/Amazon_ Machine_Image
[2] Bittorrent (Protocol), Sept. 2011. [Online]. Available: http:// en.wikipedia.org/wiki/BitTorrent_(protocol)
[3] Bloom Filter, Sept. 2011. [Online]. Available: http://en.wikipedia. org/wiki/Bloom_filter
[4] Filesystem in Userspace, Sept. 2011. [Online]. Available: http://fuse. sourceforge.net/
[5] Rabin Fingerprint, Sept. 2011. [Online]. Available: http://en. wikipedia.org/wiki/Rabin_fingerprint
[6] Reiserfs, Sept. 2011. [Online]. Available: http://en.wikipedia. org/wiki/ReiserFS
[7] Xfs: A High-Performance Journaling Filesystem, Sept. 2011. [Online]. Available: http://oss.sgi.com/projects/xfs/
[8] Data Deduplication, Sept. 2013. [Online]. Available: http://en. wikipedia.org/wiki/Data_deduplication
[9] A.V. Aho, P.J. Denning, and J.D. Ullman, ''Principles of Optimal Page Replacement,'' J. ACM, vol. 18, no. 1, pp. 80-93, Jan. 1971.
[10] 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.
[11] R. Coker, Bonnie++, 2001. [Online]. Available: http://www. coker.com.au/bonnie++/
[12] D. Eastlake, 3rd, Us Secure Hash Algorithm 1 (sha1), Sept. 2001. [Online]. Available: http://tools.ietf.org/html/rfc3174

[13] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, ''Dynamo: Amazon's Highly Available Key-Value Store,'' in Proc. 21st ACM SIGOPS SOSP, New York, NY, USA, 2007, vol. 41, pp. 205-220.

[14] 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 Proc. 7th Conf. File Storage Technol., 2009, pp. 197-210.

[15] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, ''Above the Clouds: A Berkeley View of Cloud Computing,'' Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, CA, USA, Rep. UCB/EECS 28, 2009.

[16] S. Ghemawat, H. Gobioff, and S.T. Leung, ''The Google File System,'' in Proc. 19th ACM SOSP, New York, NY, USA, Oct. 2003, vol. 37, pp. 29-43.

AUTHORS

M.SINDHUJA received B.E., Computer Science and Engineering degree from Anna university, Tiruchirapalli and currently pursuing M,E., Computer Science and Engineering degree from Arasu Engineering College, Kumbakonam, India.

R.RAJAKUMAR obtained his Master degree in Information Technology from Bharathidasan University, Tirchirappalli. Then he obtained his Master's degree in Computer Science and Engineering from PRIST University Thanjavur and now Registering PhD in Computer Science majoring in Wireless Communication from Bharathiar University, Coimbatore, India.

M.KANNAN Obtained his B.Tech, degree from Arasu Engineering College, Kumbakonam, Then he obtained his Masters's degree in Computer Science and Engineering from Anna University, Tirchirappalli Currently, he is an Assistant professor in Department of Computer Science and Engineering at Arasu Engineering College, Kumbakonam, India.

