Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 625070, 11 pages
http://dx.doi.org/10.1155/2013/625070

Research Article

IM-Dedup: An Image Management System Based on Deduplication Applied in DWSNs

Jilin Zhang, Shuting Han, Jian Wan, Baojin Zhu, Li Zhou, Yongjian Ren, and Wei Zhang

Department of Computer and Technology, Hangzhou Dianzi University, Hangzhou, Zhejiang 310018, China

Correspondence should be addressed to Jian Wan; wanjian@hdu.edu.cn

Received 9 February 2013; Accepted 18 March 2013

Academic Editor: Neal N. Xiong

In distributed wireless sensor networks (DWSNs), the data gathered by sink is always massive and consumes a lot of resources. It is suitable for cloud computing platform to apply service in data processing system. In cloud computing, IAAS platform provides services and calculation to the user through the virtual machine. The management of virtual machine images not only consumes a huge amount of storage space but also gives large pressure on network transmission. By using deduplication technology in openstack, this paper designed and implemented, an image management system IM-dedup, which uses static chunking (SC) to divide image file into blocks of data, avoid duplication data blocks transmission on network by using fingerprint pretransmission technology, and reduce storage space by deploying kernel mode file system with deduplication in the image storage server. The experimental results showed that the system not only reduced 80% usage of the virtual machine image storage, but also saved at least 30% of transmission time. Furthermore, the research on virtual machine image format showed that "VMWare Virtual Machine Disk Format" (VMDK), "Virtual Desktop Infrastructure" (VDI), "QEMU Copy On Write2" (QCOW2), and RAW image formats are more suitable for the IM-dedup system.

1. Introduction

Deduplication technology is a lossless data compression technology, which first utilizes the characteristics of the data to detect duplicated objects in data streams. After detection of duplications, it replaces the duplicated copies of the data object by using pointer to the unique data object. Deduplication technology is able to share duplicated data blocks across files, which leads to a high data compression rate, and therefore has been widely used in data backup archive field [1–5].

Gathering remote information is an important part of applications in distributed wireless sensor networks (DWSNs). For example, monitoring temperature, lightness, humidity, atmospheric pollution, building structure integrity, and chemical and biological threats. Since the data gathered by sinks is always massive and consumes a lot of resources, it is suitable to apply a cloud computing platform to host and process massive data in such a situation; the architecture is shown in Figure 1. IAAS cloud computing platforms, such as eucalyptus [6], EC2 [7], and openstack [8], use virtual machines to provide computing services for the users. Since the needs of users are flexible and different, it should support various virtual machines with a variety of configurations, such as different lineage operating system, or 32/64 bit system. In order to establish a virtual machine quickly, a cloud computing system uses virtual machine images to manage virtual machines. A single user may have multiple differently configured virtual machine images, each of which is usually larger than 1 GB. With the expansion of cloud scale, the cost of the management of virtual machine images in cloud computing systems increases fast, particularly the transmission time consumption and image storage consumption. The comparison of time consumption between MD5 and SHA1 is shown in Tables 1 and 2 gives the comparison of average execution time and CPU usage between MD5 and SHA-1.

Some recent studies [9, 10] have shown that more than 80% of the data blocks in virtual machine storage are duplicated. In other words, the 80% of the data blocks are shared by different versions of virtual machine image [11]. Therefore, the application of deduplication technology is helpful for IAAS cloud computing platforms to manage data

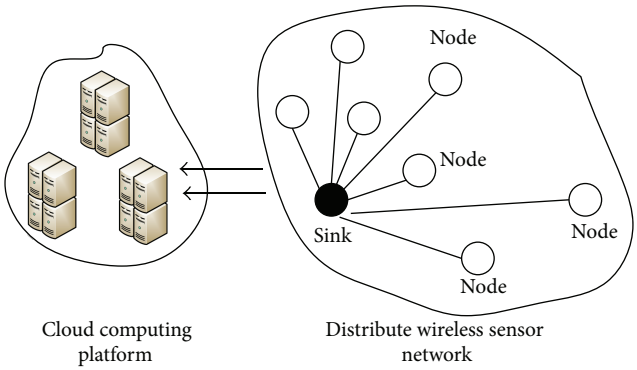

Figure 1: The architecture of data processing system in distribute wireless sensor networks.

Table 1: The comparison of time consumption between MD5 and SHA1 [27].

Hash function	1 K	100 M	1.5 G
Md5	1 ms	0.931 ms	20.762 s
SHA-1	1 ms	0.987 ms	21.518 s

Table 2: The comparison of average execution time and CPU usage between MD5 and SHA-1 [27].

Hash function	MD5	SHA-1
Average execution time	48 S	48.2 S
CPU usage	28%	72%

Test data set is vmware virtual operating system disc files, Windows 2003, CPU: P4, 28 G, AM: lG, D: 4lXBDE.

more efficiently. Most of the existing researches are concerned about how to deduplicate virtual machines, but few of them are concerned on particular open source cloud computing platforms. Some commercial file systems also can support deduplication technology, such as SDFS [12], ZFS [13], which can achieve high IO throughput. But all of them are designed for enterprise applications and require a big memory which is unsuitable for openstack.

In this paper, an image management system, named IM-dedup, is presented to solve the problems and difficulties of the openstack cloud-computing platform. The presented system applies static block technology to divide a file into blocks of data, uses the fingerprint pretransmission technology to avoid transmission of duplicated data blocks, deploys the kernel-space deduplication file system on the image storage server to reduce storage space, and uses the memory filter to reduce the overhead of disk index. Besides, it also improves the locality of data by centralizing fingerprints in disk to achieve a higher IO throughput rate with the limited memory occupancy rate.

The experiment showed that the system not only reduced at most 80% of storage capacity expenses on virtual machine image storage but also saved at least 30% of transmission time. In addition, our results on different image formats of virtual machine showed that VMDK, VDI, QCOW2, and RAW image formats are more suitable for this system.

2. Related Work

In the field of data storage and backup research, deduplication is the most important emerging technology. The complete process of deduplication consists of four steps: file chunking, fingerprint extraction, fingerprint lookup, and data storage. Researchers have made studies of deduplication technology from various perspectives, mainly focused on the following three aspects.

2.1. Studies on Strategies of Data Partitioning and Feature Extraction. AA-dedupe [14] is an application-aware deduplication technology, which applies different methods to chunk and calculate fingerprints according to file types. To improve efficiency, some excessively small files (smaller than 1 k) are ignored. SAM [15] is a semantic-aware deduplication technology. It can reduce deduplicated data blocks selectively by analyzing the feature of the file size, file location, and file type. In addition, SAM combines global-file-level deduplication with local-block-level deduplication to find the trade-off point between duplication rate and system performance.

2.2. Studies on Strategies of Memory Index Lookup. As mentioned in many papers [11, 16–18], fingerprint lookup has seriously affected IO performance in large-scale systems. Therefore, how to improve the efficiency of index query has become one of the hottest research spots in the field of deduplication technology. DDFS [11] uses Bloom filter and cache technology to improve the performance of data block lookup. Due to the use of Bloom filter, DDFS demands a big memory and thus has poor scalability, which leads to poor support for the operation of delete or update. Mark et al. [16] proposed an online block-level deduplication technology, sparse indexing, which is based on the similarity detection. This technique divides the data stream into big segments. In order to locate a similar data segment, the technology needs to extract hook fingerprints which can be shared by many blocks. The hook fingerprints of some data blocks may be the same one, and it means that these data blocks are similar to each other. It helps to reduce the sampling rate and lookup complexity. Even though sparse indexing demands less memory than DDFS, its performance of deduplication depends on the locality and sampling rate of hook fingerprint.

2.3. Studies on Transmission Strategies. The authors of paper [19] proposed a cloud backup system, called Cumulus using a simple communication protocol. It only needs to be installed in the client, and the deduplication process is also completed on client, and then Cumulus stores the data block into a remote server. However, the client must maintain a fingerprint index table in order to deduplicate between different files. Though Cumulus has improved the performance of transmission and storage, the system still has a poor scalability. LBFS [20] uses content-based chunking method to divide the files into many chunks. It transmits chunk fingerprints over the network to identify whether a data chunk is duplicated or not. By only transmitting an unduplicated data chunk, LBFS thus saves network bandwidth. The LBFS

runs in user space. Though the LBFS saves bandwidth during transmission, the received files in the storage system are still full-size without any deduplication treatment.

Semantic-aware and application-aware deduplication technology can achieve a tradeoff between deduplication rate and system performance, but this technology is more suitable for personal file backup rather than for virtual machine image management. Enterprise-class deduplication file systems such as DDFS, ZFS, and SDFS demand enterprise-class equipments, while Openstack is usually deployed on limited hardware resources, thus enterprise-class deduplication file systems are not suitable for openstack obviously.

According to the problems and difficulties encountered in image management of Openstack, this paper presented a novel image management system, called IM-dedup system, which uses the static block technology to divide a file into a number of blocks, uses fingerprint pretransmission technology to avoid the transmission of duplicated blocks, deploys the kernel file system with deduplication functionality in the image storage to reduce storage space, uses memory filter to reduce the number of disk indexes, centralizes fingerprint storage area to improve the locality of data accessing, and uses limited memory to achieve a higher IO throughput rate.

The paper is organized as follows. In Section 3.1, the openstack system is introduced. The detailed analysis of image storage and the deduplication process on a sever are given in Section 3.2. In Section 3.3, the IM-dedup system is described and analyzed. In Section 4, the system evaluation methods and results are given. Finally, some conclusions are drawn in Section 5.

3. Design and Implementation of IM-Dedup System

This system takes the advantages of source-deduplication saving bandwidth and target-deduplication saving storage space. It chunks the file data for only once and computes the fingerprint for just once and thus avoids repetitive computation and fully utilizes the existing resources.

3.1. Openstack. Openstack is an open source project developed by NASA and Rackspace, which can be used to build private and public clouds. And it consists of several key components: nova, glance, keystone, swift, network, and so forth. Each component communicates with others by a message queue. Nova is the computation component, which provides application, creation, activation, and destruction services for the virtual machine. Glance is the image management component, which provides storage, query, and registration services for images. Keystone provides registration, management for the account. Swift is the object storage component for Openstack, which provides high-reliability and high-scalability storage service. Network component manages each node's network connection, it can avoid network's bottleneck effectively and improve efficiency. Figure 2 shows the architecture of Openstack.

At present, there are two ways to add an image into openstack.

(1) By glance client and by executing glance add command, the virtual machine image will be added into the storage backend of glance.

(2) By dashboard, users can take a snapshot of a virtual machine and save it as an image.

Method (1) is mostly used by an administrator to manage public virtual machine images.

Method (2) is mostly used by users to manage private virtual machine images.

Every image uploaded to the glance will be assigned the UUID randomly, which is the only identification in the system. No matter which method used, there are two things that openstack must do when storing image:

(1) store a virtual image in the glance,

(2) store the metadata in the glance, which includes registration of the image information in the database, and then mark the image state as saved.

The images of Openstack have 6 states, including queued, saving, active, killed, deleted, and pending_delete. Queued state represents that the UUID of an image is registered into glance, but no image data has been uploaded. Saving state represents that the image data is being uploaded. Killed state represents that there is something wrong during uploading. Active state represents the availability of the image. Delete state represents that image data will be cleaned later; however, the image still can be used at present. Pending_delete state is similar to delete state, but image data can be to reinstated.

Since the needs of users are various, there will be a number of virtual machine images of different versions. As a result, huge storage space is required to store them, and considerable time is needed to transmit them over networks.

3.2. The Process of Server-Side Image Deduplication and Storage. The Openstack should be deployed on the Ubuntu Linux kernel, which uses EXT3 file system [21] to manage files and devices. The hard disk partition of EXT3 file system consists of a series of 4096-byte (4 KB) data blocks; these data blocks are divided into several block groups which are of the same size. Each block group contains a bitmap block (used to record the use of data blocks in the block group), an inode linked list (records each file's inode) and an inode bitmap block (records the use of inode). Each file has a 128-byte inode, which records the creation, modification, access time, size, and save location information of the file. As shown in Figure 3, in the Linux file system, each file will use the data structure called inode, which describes a file or a directory. Each inode contains all of the description information of a file or a directory, including file type, access permissions, file owner, timestamps, file size, data block pointers, and so on. The data pointer can be divided into three types: direct pointers (or the single-level pointers), the second-level pointers, and the third-level pointers. They point to the data from different levels. The three types of pointers are shown in Figure 4.

In order to implement the deduplication and reduce the local storage space of the images, the liveDFS [22] data

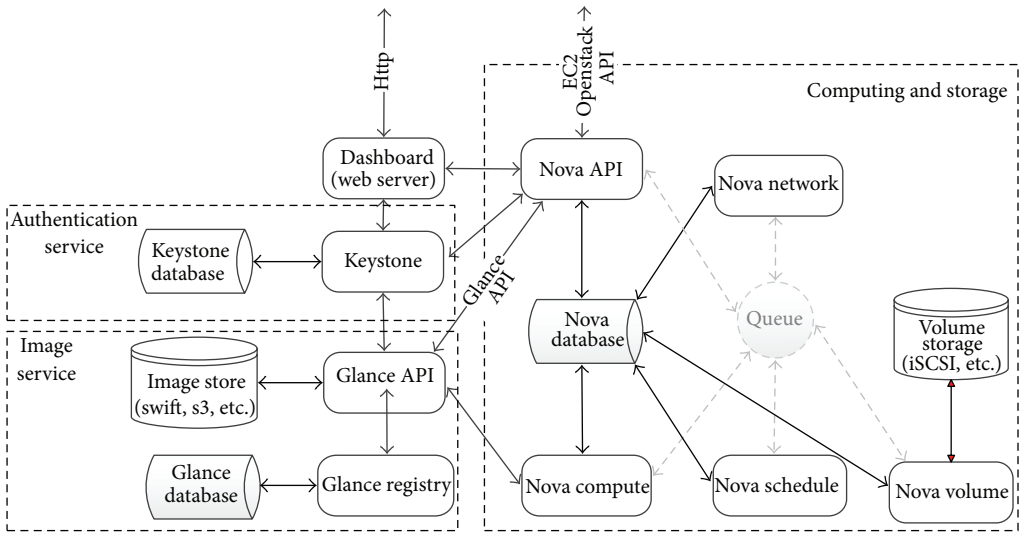

Figure 2: The architecture of Openstack.

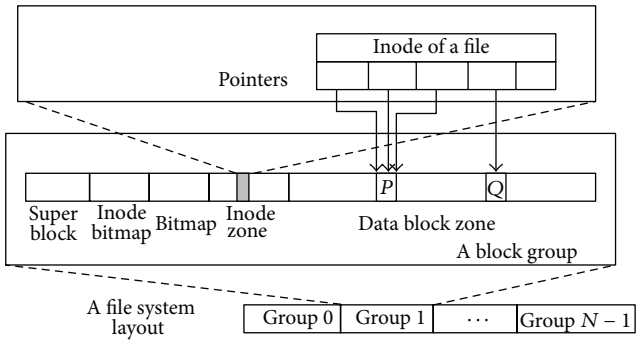

Figure 3: The layout of the file system on the disk [22].

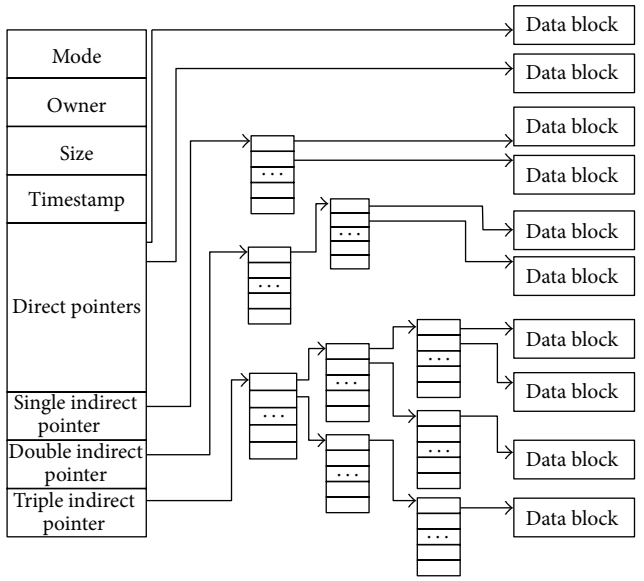

Figure 4: The architecture of an inode in EXT3 file system.

deduplication method is used into the image data storage system on a single computer node. The file system uses a method of fixed block size which can be demonstrated to be capable of optimizing the performance of the CPU and memory. In order to be compatible with liveDFS file system, the IM-dedup system also chooses to use the method of fixed block size. It means that IM-dedup system also divides a file into a number of chunks which size is 4 KB. And the system will extract fingerprint values in a unit of 4 KB. And the system will extract fingerprint values in a unit of 4 KB. For example, file F_1 is cut into C_1, C_2, and C_3, and then the fingerprint values are f_1, f_2, and f_3, respectively; the data blocks C_1, C_2, and C_3 are stored in the file system. F_2 is cut into C_4, C_5, and C_6, and the fingerprint values are f_4, f_5, and f_6, respectively. Finding and comparing through the fingerprint values, if $f_1 = f_5$, then we only stored C_4 and C_6, and let the fingerprint point to C_1, and thus two pointers pointing to one data block, as shown in Figure 5.

3.2.1. Fingerprint Storage. Ext3 file system manages the disk by block groups, and the size of group is 128 MB. And also each file is cut into blocks which size is 4 KB. So, each block group has 32768 blocks in total, and it occupied 655360 bytes space in fingerprints storage, equivalent to the 160 blocks. These 160 blocks exist in the front space of each block group called fingerprints storage area. The fingerprints storage area consists of an array of fingerprints and the reference counter, and it can index to the data block in every block group through each disk data blocks block number (block address) of the same block group, as shown in Figure 5.

One of the fingerprints contains 20 byte, the former 16 byte record fingerprint values of a block and the later 4 byte record the reference number, as shown in Figure 6.

3.2.2. Fingerprint Filter. The fingerprint filter is the index structure in memory. It aims to accelerate the searching speed

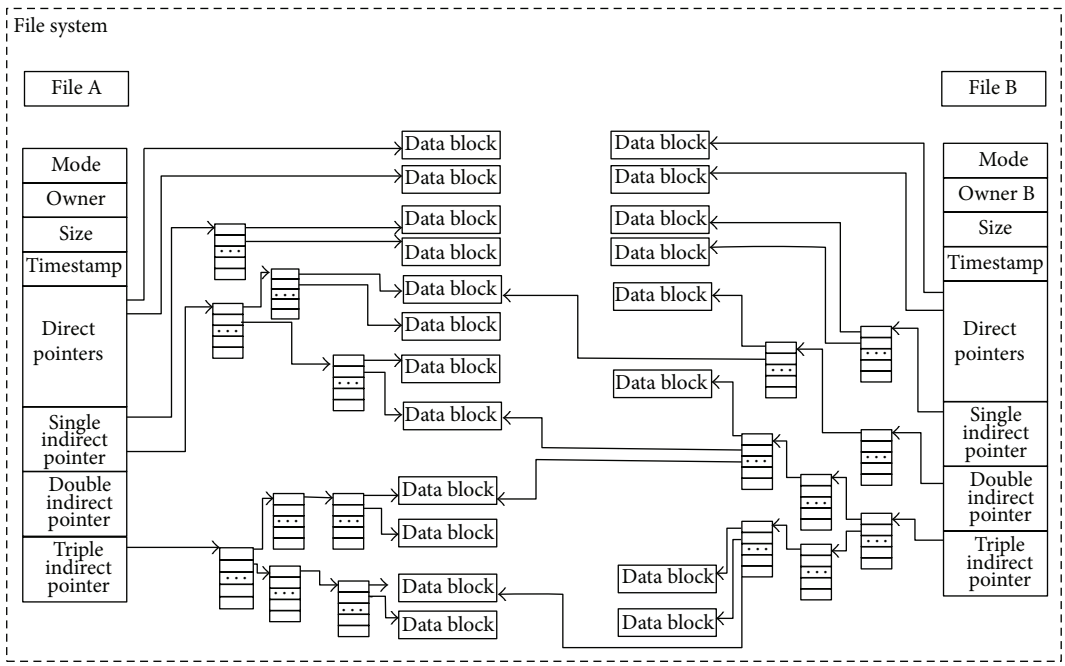

Figure 5: Multiple files sharing data block.

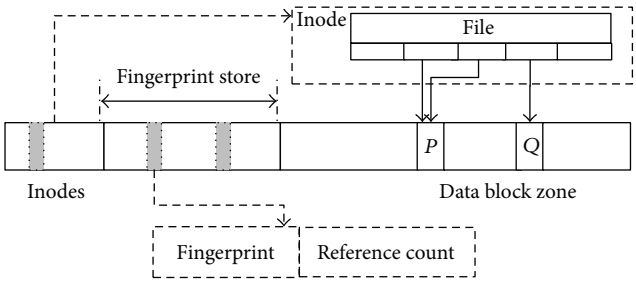

Figure 6: The layout of the a block group [22].

of the fingerprint on disk. Figure 7 describes the fingerprint filter design in detail. The fingerprint filter is the second-level filter. The first-level filter is named index key, which maps the former n bit of fingerprint. The second-level filter is named bucket key, which maps the later k bit of fingerprint.

When initializing the fingerprint filter, Mount program allocates index key table firstly. The index key table is an array of 2^n elements, each element points to the head pointer of a bucket linked list. The tuples which are the same to index key are saved in the bucket which is pointed by this index key. Bucket line is combined with tuples bucket key block number, if one bucket is full, the system will create a new bucket to build bucket linked list. In order to use binary search method to query the bucket key efficiently, we sort the elements in each bucket by bucket key. It needs to be emphasized that the fingerprint filter initialization is a one-time process, and the cost of theinitialization depend on the total amount of data. Before a new block of data is written to the disk, the fingerprint filter is firstly queried. If the new blocks fingerprint is matched with the former $n + k$ bit of the filter, then the filter will return the corresponding data block number. The fingerprint values which share the same prefix "$n + k$" may be more than one, and the filter may return a number of block numbers. In order to eliminate false positives, the system will find the corresponding fingerprint storage area through the return block number to prove if the fingerprint values in data block are exactly matched with the one in fingerprint storage area. After completion of each operation (write, modify, delete), the system will update the fingerprint filter. Ng et al. [22] proved that when $n = 19$ and $k = 24$, the efficiency of the filter can achieve a very good performance through mathematical calculations.

Figure 8 is a new process of writing files.

When a new file comes, system will first chunk it into blocks. And then, the fingerprint calculation program will calculate the fingerprint. Once a fingerprint is produced, it will be transferred into fingerprint filter. If the fingerprint is not in the filter, it indicates that the block is not duplicated. Then the system will write the block into disk and then modify the inode. If the fingerprint is in the filter, the program will start another process into disk to search the fingerprint. If fingerprint is not in the disk, it indicates that the block is a new one. It should be stored into disk, and then some modification should be done to the inode. If fingerprint is in the disk, which means the block is a duplicated one, the block data should be dropped, and what the program needs to do is to modify the pointers in the inode.

3.3. The Design of IM-Dedup System. The infrastructure of IM-dedup system is shown in Figure 9. In Openstack, nova integrates physical resources from compute nodes and provides users computing services. Glance manages the storage, retrieval, and registration of virtual machine image. When a user makes a launch request, nova sends a retrieval request.

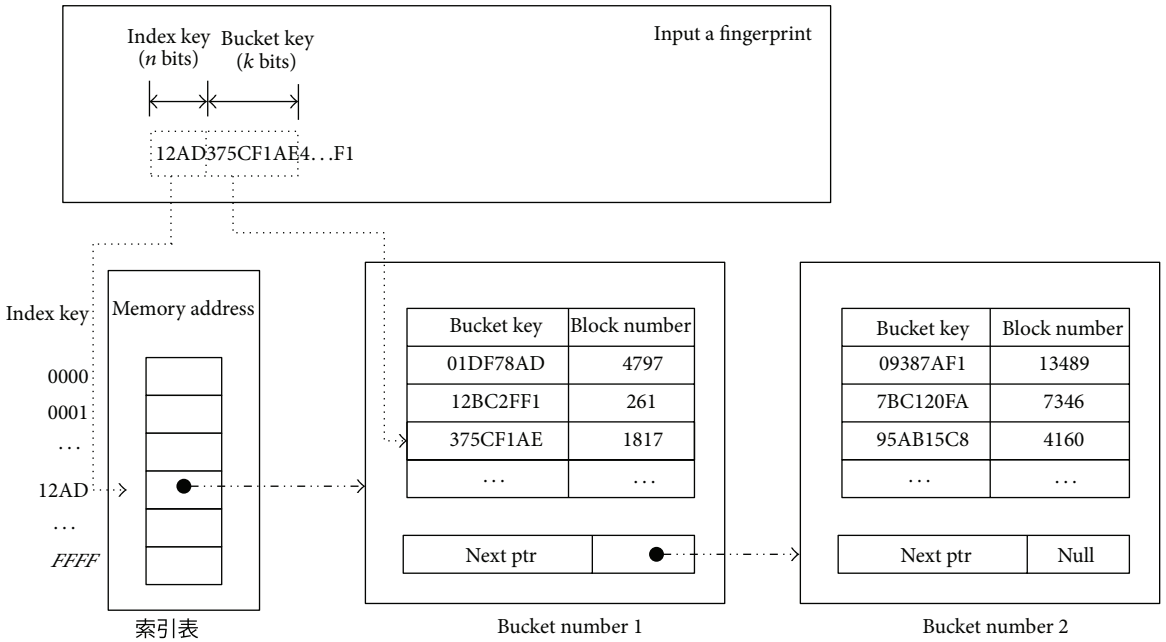

Figure 7: The structure of fingerprint filter [22].

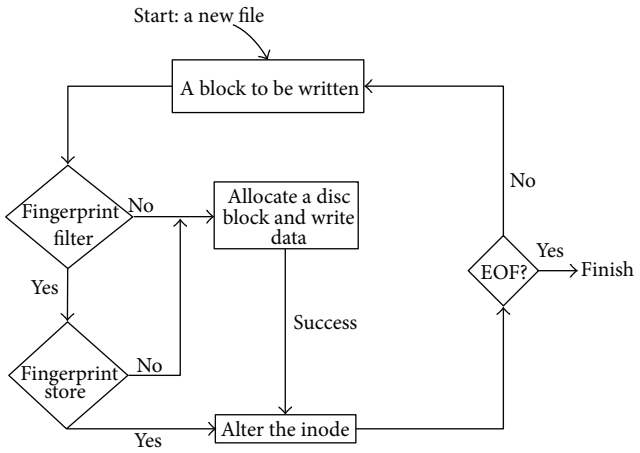

Figure 8: The judging procedure of write data blocks.

And then glance delivers the requested image to a compute node which will then launch this virtual machine image.

When a glance client needs to upload an image, the glance server will register the image information in the registry server and then uploads the image to the storage backend.

3.3.1. The Process of IM-Dedup System. Figure 10 describes the workflow of the process of the image uploading.

Step 1. The glance client sends an image write request to the glance server. The glance server registers the image information to the image database. And the glance server enters the file write mode, makes a new inode, and prepares for the file writing.

Step 2. The glance client starts the slice operation. After getting a chunk c, glance client computes the related fingerprint f and sends it to the glance server.

Step 3. The glance server receives the fingerprint f and then sends it to the fingerprint filter. The fingerprint filter checks f to make sure the $n + k$ bit of f in the filter. If not in, go to Step 6; if in, go to Step 4.

Step 4. Start the process of fingerprint storage retrieval. If f is a new fingerprint, go to Step 5; if not, go to Step 6.

Step 5. If there is f of chunk c the same as chunk c fingerprint f. Modify the present inode pointer to point to c.

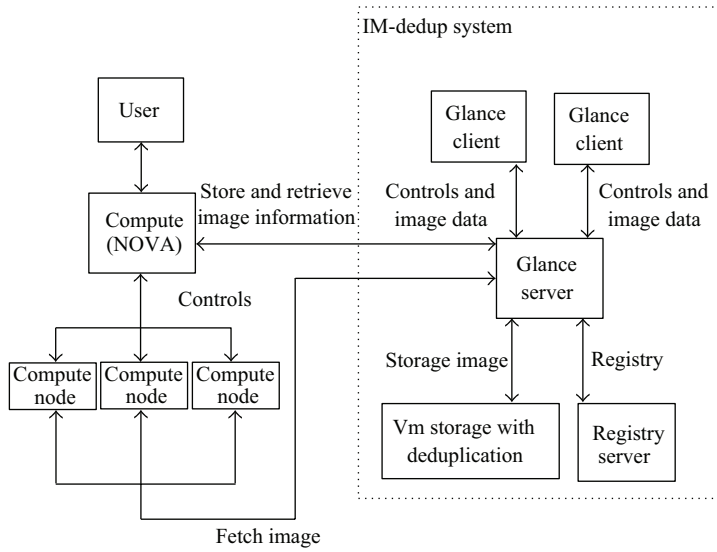

Figure 9: The architecture of IM-dedup system with Openstack.

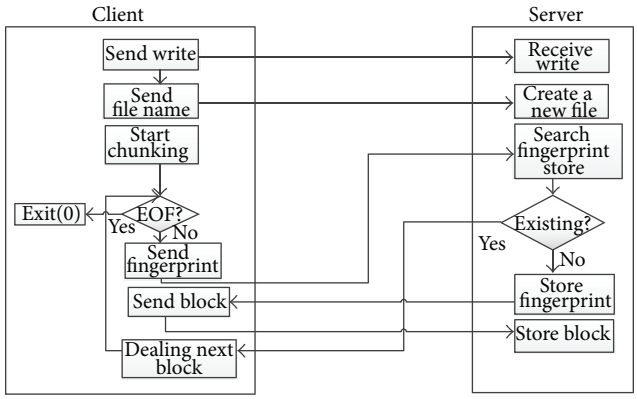

Figure 10: The process of IM-dedup system.

Step 6. Return a request for chunk c to glance client, and then store chunk c onto storage backend, and modify the present pointer to point to chunk c.

Step 7. Go to Step 2, until the end of file.

Step 8. Finish image upload process, and mark it as active.

3.3.2. Chunking. At present, there are two kinds of file-slice methods, including static chunking [23] (SC, as is used in venti [24] and oceanstore [25]) and content defined chunking [11] (CDC, as is used int LBFS [20] and Pastiche [26]). And the deduplication ratio is higher by CDC than by SC [23]. And fingerprint extracting methods widely used are still hash algorithms, like MD5, SHA-1, and Rabin hash.

The present image deduplication technologies only handles single node deduplication, because t the overhead of network data transportation are not considered. To solve this problem, it is necessary to slice the image in the glance client. The present image deduplication technologies only handles single node deduplication, because t the overhead of network data transportation are not considered. Therefore, IM-dedup system adopts SC as the slice method. For seamless combination of native deduplication on image storage server and remote slicing to avoid redundant slicing and fingerprint calculating, IM-dedup system sets chunk size as 4096 bytes.

3.3.3. Methods for Calculating the Data Fingerprint. Each file slicing needs to calculate the data fingerprint. And the system then sends the fingerprint and other metadata to the image storage management server. And on image storage backend, the system will retrieve the fingerprint to make sure that if it is necessary to send this data to image storage backend.

At present, MD5 and SHA-1 are the two most widely used hash algorithms. MD5 groups the input as 512 bits, and the output is 4 32-bits cascades which makes up a 128-bits information abstract. And MD5 is one of the safest encryption algorithms. And SHA-1 generates 160-bits information abstract based on MD5. SHA-1 is a widely used hash algorithm and also the one of most advanced encryption technologies.

4. Experiments

4.1. Experimental Setup. The Openstack Image Service provides discovery, registration, and delivery services for disk and server images. IM-dedup system can do more than Openstack Image Service can. IM-dedup system not only can provide discovery, registration, and delivery services but also provide deduplication and network transmission optimization. The experiments are designed to answer the following 3 questions.

(1) How much storage space saved by IM-dedup system compared to Openstack Image system?

(2) How much transmission time can IM-dedup system save compared to Openstack Image system?

(3) What is the most efficient image formats according to IM-dedup system? In other words, what image formats can reach the highest deduplication rate and get the best transmission optimization results.

In our experimental setup, IM-dedup system is deployed on Lenovo R510 G7 1U server which has an Intel Quad-Core E5606 CPU 2.13 GHz 8 M cache, R-ECC DDR3-1333 4 G memory SAS2.5 inch 300 G disk. The nodes are connected by 1 Gb/S independent Ethernet network.

There are three servers. The first one is used as deduplication image storage server deployed with IM-dedup server. The second one is used as original image storage server deployed with glance server. The last one is used as client, which is in charge of uploading virtual machine image data to servers.

4.2. Data Set. Openstack provides a multiformat image registry; the image service allows uploads of private and public images in a variety of formats, including

(i) Raw (img),

(ii) Machine (kernel/ramdisk outside of image, aka AMI),

(iii) VHD (Hyper-V),

(iv) VDI (VirtualBox),

(v) qcow2 (Qemu/KVM), VMDK (VMWare),

(vi) OVF (VMWare, others).

Some researcher found that the format of virtual machine image impacts a lot on deduplication rate [9]. Images in our data set adopt standardized configuration (2 G memory, 10 G harddisk). Table 3 shows 35 virtual machine images, which contains 7 operating systems and 5 formats. Some of them are in the same lineage (such as Centos.6.2 and Centos.6.3), some of them are in different branches of linux operating system (such as Centos.6.2 and Ubuntu 11.10); some of them are absolutely different (such as Win7 and Centos.6.3). All of them are mainstream operating systems.

4.3. Deduplication Rate. Deduplication rate is an important indicator to the effect of deduplication, which is defined as follows:

$$\text{deduplication rate} = \frac{\text{original bytes of disk image} - \text{stored bytes of disk image}}{\text{original bytes of disk image}}. \quad (1)$$

Deduplication rate depends primarily on deduplication algorithm and deduplication data sets. As some researches [14, 15, 19, 23] show that when the granularity of deduplication increases, the deduplication rate will decrease. But the granularity deduplication decreases, the IO throughput will increase. And it will also reduce CPU efficiency, enhance the memory demand, and decrease overall system performance. At present, the widely adopted deduplication granularity is 8 k. There are some deduplication systems that use 4 M chunking granularity.

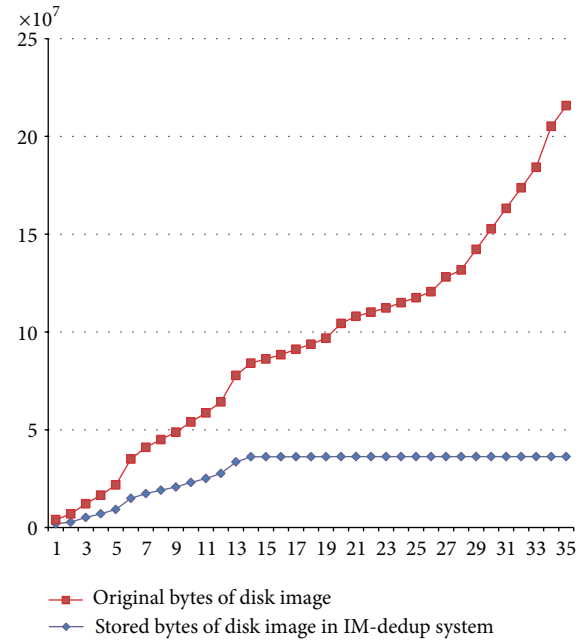

Figure 11: Accumulated storage utilization.

In addition, deduplication rate is also impacted by data sets. For the same data set, the higher deduplication rate the better. In order to test the deduplication performance of IM-dedup system image storage server, this paper uses the original Openstack glance storage backend EXT3 file system as the benchmark.

The accumulated storage utilization of the virtual machine image in the original storage system is shown in the Figure 11. In comparison, when we use the IM-dedup system, with the increase of the number of images, the growth of the storage space slowly and finally leveled off. The reason is that the experimental data sets only using seven image versions (they are Centos.6.2, Centos.6.3, Ubuntu 11.10, Ubuntu 12.04, Ubuntu12.10, WinPro, and Win7), and each image was made into five formats. Although there are 35 images, in fact there are only 7 versions of images, different formats of an operating system may share a plenty of data blocks. Therefore, the deduplication system performs efficiently. The curve of IM-dedup system growth almost decreases to zero. Ideally, if the server stored enough virtual machine image versions, the hit rate of duplicated block will not grow in the server. Because the image server does not need to store duplicated block, the space occupied by virtual machine image will not grow. We can learn from Figure 11 that deduplication rate can be up to 80%.

In order to test the different image formats deduplication rate in the system, this paper selects different system virtual machine images, classified by format, divided into five categories, as shown in Figure 12.

Figure 12 shows that with the same image configuration, raw format occupied the maximum disk space, vmdk, VHD, VDI, and Qcow2 occupied disk space closely. But VHD format image deduplication rate is far lower than the average

Table 3: Images size in the data set (kB).

	Centos. 6.2	Centos. 6.3	Ubuntu 11.10	Ubuntu 12.04	Ubuntu 12.10	Win Pro	Win 7	Totol (kB)
VMDK	2111684	2103556	2768580	2549572	3040452	3587332	7507908	23669084
VHD	2171436	2161196	2853588	2630300	3111700	3615632	7591788	24135640
VDI	2138160	2132016	2799664	2578480	3068976	3612720	7565400	23895416
QCOW2	2107276	2099788	2765772	2546956	3036748	3583940	7507012	23647492
RAW	2100632	2091528	2753424	2536012	3022748	10485764	20971524	43961632
Total (KB)	10629188	10588084	13941028	12841320	15280624	24885388	51143632	

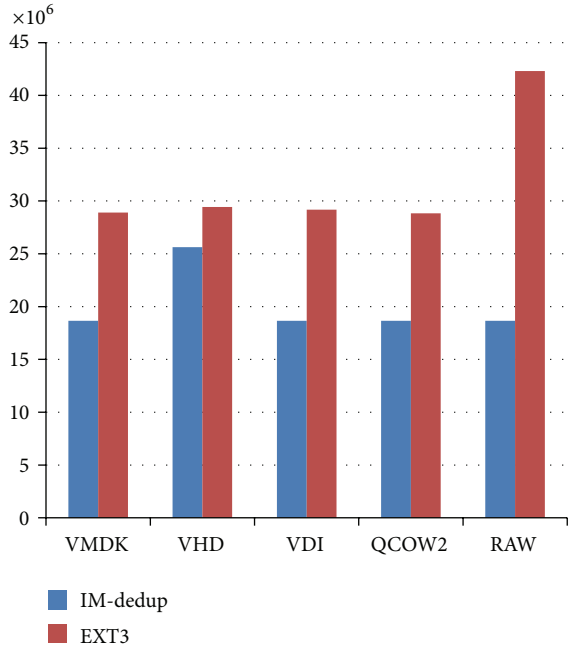

Figure 12: In contrast of the IM-dedup system deduplication effect of virtual machine images of different formats.

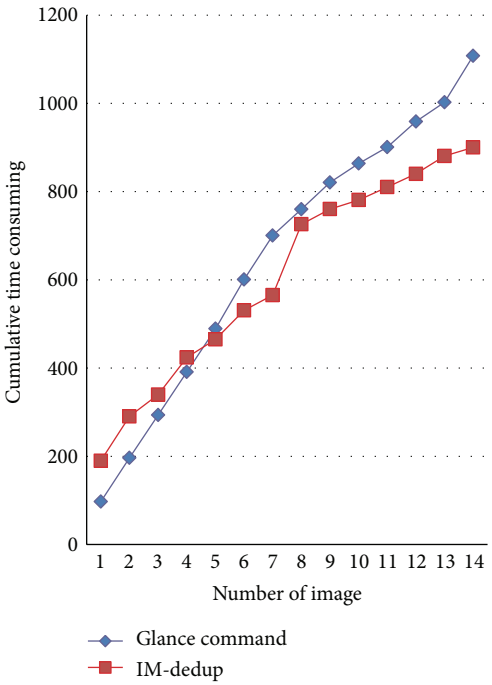

Figure 13: Cumulative time-consuming comparison between glance and IM-dedup system.

deduplication rate 36%. It has only about 13%. Although they have the same conditions on the virtual machine operating system configuration, there are still significant differences between the deduplication rate. This attributes to the internal disk layout of VHD format. But in this paper, there is no in-depth research in this aspect. Experiments show that the openstack supported image formats, VMDK, VDI, QCOW2, and RAW, is ideal for IM-dedup system management, especially the RAW format. Deduplication rate can reach 56%. In practice, the users of the Openstack should recommend the use of these four forms of virtual machine images.

4.4. Network Transfer Optimization. IM-dedup system uses sockets to implement chunk data blocks transmission. With the method of the fingerprint pretransmission, IM-dedup system avoids the transmission of duplicated data blocks. Openstack uses the glance add command to add a new image into image system. Figure 13 shows the time-consuming comparison between glance and IM-dedup system. It can be seen from the chart: when uploading a few images (less than 5 in this experiment), glance performs better than IM-dedup. That is, because glance uploads a whole image without the overhead of chunking, calculation fingerprint and fingerprint index. And there are few images in the image server, so few chunks can be deduplicated. In comparison with glance, IM-dedup system seems to have larger overhead.

However, as the number of images increases (more than 5 in this experiment), the slope of the curve decreases, and cumulative time consuming of IM-dedup system is less than glance. That is, because the number of duplicated data blocks increases as the number of images in storage server increases, and thus more data blocks do not need to be transmitted in IM-dedup.

We also noticed that cumulative time consuming grows very fast when 8th image was being uploaded. Then, we checked the data set and found that the former 7 images are Linux branches (such as Centos, Ubuntu). The 8th image is win7, which shares little duplicated data block with Centos and Ubuntu. The result is also proved in [9].

When the number of upload image grows further, the slope of the IM-dedup curve decreases; however, the slope of

glance does not decrease. In the ideal state, deduplication rate is 100%, and the slope of IM-dedup is infinite toward zero. But in real case, the transmission of requests and fingerprints cannot be neglected. In the best case, we can expect that the time is all consumed by the requests and fingerprints.

Our lab environment is equipped with 1 GB/S Ethernet, which is much better than general conditions. The cost of adding an image by original methods depends on the image size and network condition. When network bandwidth is low, it will consume considerable time. In which case, the advantages of IM-dedup network transport system are more obvious.

After all, the performance of IM-dedup is more prominent when the image number is large and in a low bandwidth environment.

5. Conclusions and Future Work

In this paper, the IM-dedup system based on image management system of deduplication was presented, which could be used in data processing system of distributed wireless sensor networks. The IM-dedup system combined deduplication technology with the original basic functions of openstack, such as glance image upload and registration, to achieve double deduplication of local storage and network transmission. By replacing the original image management system glance in openstack, it achieves better performance. The experimental results showed that IM-dedup system saved at most 80% of the storage space and 30% of image upload time in the image storage server.

This paper mainly studied the way to reduce network transmission time in image upload and achieve deduplication technology in image storage. Future work will be done on the further optimization of image download process by applying deduplication technology.

Acknowledgments

This paper is supported by State Key Development Program of Basic Research of China under Grant no. 2007CB310900, the Hi-Tech Research and Development Program (863) of China under Grant no. 2011AA01A205, Natural Science Fund of China under Grant no. 61202094, no. 61003077, no. 60873023, and no. 60973029, the Science and Technology Major project of Zhejiang Province (Grant no. 2011C11038), Zhejiang Provincial Natural Science Foundation under grant no. Y1101104, Y1101092, and Y1090940. Zhejiang Provincial Education Department Scientific Research Project (no. Y2-01016492).

References

[1] L. Youyou, A. Li, and S. Jiwu, "A remote data backup system with deduplication," *Journal of Computer Research and Development*, vol. 2012, S1, pp. 206–210, 2012.

[2] L. Qiang, Z. Ligu, and Z. Fu, "The desktop backup technology based on cloud storage," in *Proceedings of the China National Computer Conference (CNCC '10)*, CCF, HangZhou, China, 2010.

[3] M. Jian-ting and Y. Pin, "Deduplication-based file backup system for multiuser," *Computer Engineering and Design*, vol. 32, no. 11, pp. 3586–3589, 2011.

[4] W. Zhen, *The research on deduplication software technology in backup system of bank [Ph.D. thesis]*, FuDan University, Shanghai, China, 2009.

[5] Y. Lawrence and K. Christos, "Evaluation of efficient archival storage techniques," in *Proceedings of the Conference on Mass Storage Systems and Technologies (NASA/IEEE MSST '04)*, pp. 227–232, Greenbelt, Md, USA, 2004.

[6] Eucalyptus project, 2012, http://www.eucalyptus.com/.

[7] EC2, 2012, http://aws.amazon.com/ec2/.

[8] Openstack project, 2012, http://www.openstack.org/.

[9] J. Keren and L. Ethan Miller, "The effectiveness of deduplication on virtual machine disk images," in *Proceeding of the Israeli Experimental Systems Conference (SYSTOR '09)*, Article 7, New York,USA, 2009.

[10] A. Liguori and E. Van Hensbergen, "Experiences with content addressable storage and virtualdisks," in *Proceedings of the Workshop on I/O Virtualization (WIOV '08)*, San Diego, Calif, USA, 2008.

[11] Z. Benjamin, L. Kai, and P. Hugo, "Avoiding the disk bottleneck in the data domain deduplication file system," in *Proceedings of the Conference on File and Storage Technologies (FAST '08)*, San Jose, Calif, USA, 2008.

[12] OpenDedup, 2012, http://code.google.com/p/opendedup/downloads/detail?name=SDFScture.pdf.

[13] OpenSolaris, 2012, http://hub.opensolaris.org/bin/view/Community+Group+zfs/dedup.

[14] Y. Fu, H. Jiang, N. Xiao, L. Tian, and F. Liu, "AA-Dedupe: an application-aware source deduplication approach for cloud backup services in the personal computing environment," in *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER '11)*, pp. 112–120, Austin, Tex, USA, September 2011.

[15] Y. Tan, H. Jiang, D. Feng, L. Tian, Z. Yan, and G. Zhou, "SAM: a semantic-awaremulti-tiered source de-duplication framework for cloud backup," in *Proceedings of the 39th International Conference on Parallel Processing (ICPP '10)*, pp. 614–623, San Diego, Calif, USA, September 2010.

[16] L. Mark, K. Eshghi, D. Bhagwat, V. Deolalikar, G. Trezis, P. Camble et al., "Sparse indexing: large scale, inline deduplication using sampling and locality," in *Proceedings of the 7th USENIX Conference on File and Storage Technologies (FAST '09)*, pp. 24–27, San Francisco, Calif, USA, 2009.

[17] S. Rhea, R. Cox, and A. Pesterev, "Fast, inexpensive content addressed storage in Foundation," in *Proceedings of the USENIX Annual Technical Conference on Annual Technical (ATC '08)*, pp. 143–156, Berkeley, Calif, USA, 2008.

[18] 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 Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS '09)*, pp. 237–245, London, UK, September 2009.

[19] V. Michael, S. Savage, and G. M. Voelker, "Cumulus: file system Backup to the Cloud," in *Proceedings of the Conference on File and Storage Technologies (FATS '09)*, San Francisco, Calif, USA, 2009.

[20] M. Athicha, B. Chen, and D. Mazières, "A low-bandwidth network file system," in *Proceedings of the 18th ACM Symposium on*

Operating Systems Principles (SOSP '01), New York, NY, USA, 2001.

[21] G. XiMei, *An in-depth research on filesystem and disk management mechanism [Unpublished M.Phil thesis]*, Nanjing University of Aeronautics and Astronautics, Jiangsu, China, 2002.

[22] C.-H. Ng, M. Ma, T. -Y. Wong, P. P. C. Lee, and J. C. S. Lui, "Live deduplication storage of virtual machine images in an open-source cloud," in *Proceedings of the ACM/IFIP/USENIX 12th International Middleware Conference*, Lisboa, Portugal, 2011.

[23] D. Meister and A. Brinkmann, "Multi-level comparison of data deduplication in a backup scenario," in *Proceedings of the Israeli Experimental Systems Conference (SYSTOR '09)*, Article 8, May 2009.

[24] Q. Sean and D. Sean, "Venti: a new approach to archival storage," in *Proceedings of the Conference on File and Storage Technologies (FAST '02)*, San Jose, Calif, USA, 2002.

[25] J. Kubiatowicz, D. Bindel, Y. Chen et al., "OceanStore: an architecture for global-scale persistent storage," in *Proceedings of the 9th Internatinal Conference Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, pp. 190–201, Cambridge, Mass, USA, November 2000.

[26] L. P. Cox, C. D. Murray, and B. D. Noble, "Pastiche, making backup cheap and easy," in *Proceedings of the 5th Symposium on Operating System Design and Implementation*, Boston, Mass, USA, 2002.

[27] X. Shushen, *The research and application with hash algorithm [Unpublished M.Phil thesis]*, Hubei University of Technology, Hubei, China, 2006.