Indexes for Distributed File/Storage Systems as a Large Scale Virtual Machine Disk Image Storage in a Wide Area Network

Keiichi Shima
IIJ Innovation Institute
Chiyoda-ku, Tōkyō 101-0051, Japan
Email: keiichi@iijlab.net

Nam Dang
Tokyo Institute of Technology
Meguro-ku, Tōkyō 152-8550, Japan
Email: namd@de.cs.titech.ac.jp

***Abstract*—In this paper, we will show throughput measurement results of I/O operations of Ceph, Sheepdog, GlusterFS, and XtreemFS, especially when they are used as virtual disk image stores in a large scale virtual machine hosting environment. When used as a backend storage infrastructure for a virtual machine hosting environment, we need different evaluation indexes other than just I/O performance since the number of machines are huge and latency between machines and storage infrastructure may be longer in such environment. The simple I/O performance measurement results show that GlusterFS and XtreemFS perform better in read operation cases than Ceph and Sheepdog even though both are supported by the virtualization software (QEMU/KVM) natively. For write operation cases Sheepdog performed better than others. This paper introduces two indexes, $ParallelismImpactRatio$ and $LatencyImpactRation$, to evaluate robustness against the number of virtual machines in operation and network latency. We found that GlusterFS and XtreemFS are more robust against both increasing number of virtual machines and network latency than Ceph and Sheepdog.**

I. Background

The hosting service is one of the major Internet services which has been provided by many Internet service providers for long time. The service was originally using physical computers located in a datacenter to host services of their customers, however, the virtualization technology changed the hosting infrastructure drastically. Many hosting services, except some mission critical services or services which require high performance, are now providing their services using virtual host computers.

The important point of virtualization in service operation is that the efficiency in multiplexing computing resources. We want to put as many virtual machines as possible as long as we can keep the service level agreement. A virtual machine migration technology contributes this goal. If we have multiple virtual machines which does not consume a lot of resources in multiple physical machines, we can aggregate those virtual machines to fewer number of physical machines using a live migration technology without stopping them.

Usually, such a migration operation is utilized within a single datacenter only, however, considering the total efficiency of service provider operations, we need global resource mobility strategy among multiple datacenters ([1], [2], [3]). One of the essential technologies to achieve this goal is a storage system for virtual machines. When a virtual machine moves from one physical computer to another computer, both physical computers must keep providing the same virtual disk image to the virtual machine. We are typically using NFS or iSCSI technologies at this moment, however it is difficult to design a redundant and robust storage infrastructure in a wide area service operation. Recent research papers show that distributed storage management systems ([4], [5], [6], [7]) are becoming mature. We are now at the stage to start considering other choices when designing a virtual storage infrastructure.

When considering distributed storage systems, just focusing on the I/O performance is not enough, if we use them as backend storage systems for virtual machines. We need evaluation indexes which are suitable to the virtualization environment.

In this paper, we investigate the possibility and potential of recent distributed file/storage systems as virtual machine backend storage systems and evaluate their I/O throughput in a realistic large scale storage infrastructure which consists of 88 distributed storage nodes. We then define two evaluation indexes for the virtualization environment; *ParallelismImpactRatio* and *LatencyImpactRatio*. We evaluate impact to the throughput which may be affected by the number of virtual machines running in parallel and by the network latency assuming that virtual machines are operated in multiple datacenters geographically distributed.

II. Target Systems

A. Virtualization

The virtual machine hosting environment we used in this measurement was *QEMU* ([8], [9]) and *KVM* ([10], [11]). The host operating system was Linux (Ubuntu 12.04.1 LTS) and the version of QEMU/KVM software was 1.0 (the *kvm* package provided for the Ubuntu 12.04.1 LTS system).

B. File/Storage Systems

We selected the following four different stores in this experiment, with the first three belonging to the hash-based

distribution family, and the last one belonging to the metadata-based family. The reason of the choice is that these four systems are well-known and the implementation of these file/storage systems are publicly available. We are focusing on the performance of real implementations in this paper.

- Ceph
  Ceph ([4], [12]) is a distributed object-based storage system designed to be POSIX-compatible, and highly distributed without a single point of failure. On March 2010, Ceph was merged into Linux kernel 2.6.34, enabling easy integration with popular Linux distributions, although the software is still under development. Ceph provides multiple interfaces to access the data, including a file system, *rbd*[1], *RADOS* [13], and C/C++ binding. In this experiment, instead of the file system interface, we utilized the rbd interface, which is also supported by QEMU. This rbd interface relies on *CRUSH* [14], a Replication Under Scalable Hashing [15] variant with the support of uniform data distribution according to device weights.
- Sheepdog
  Sheepdog ([5], [16]) is a distributed object-based storage system specifically designed for QEMU. Sheepdog utilizes a cluster management framework such as *Corosync* [17] or *Apache ZooKeeper* [18] for easy maintenance of storage node grouping. Data distribution is based on the consistent hashing algorithm. QEMU supports sheepdog's object interface natively (the *sheepdog* method[2]) similar to Ceph's rbd interface.
- GlusterFS
  GlusterFS ([6], [19]) is a distributed filesystem based on a stackable user space design. It relies on consistent hashing to distribute the data based on a file name. The data is stored on a disk using native formats of a backend storage node. One of the interesting features of GlusterFS is that metadata management is fully distributed, therefore there is no special nodes for metadata management. GlusterFS provides POSIX semantics to its client nodes.
- XtreemFS
  XtreemFS ([7], [20]) is an open source object-based, distributed file system for wide area network. The file system replicates objects for fault tolerance and caches data and metadata to improve performance over high-latency links. XtreemFS relies on a centralized point for metadata management and for data access. The file system provided by XtreemFS is guaranteed to have POSIX semantics even in the presence of concurrent access.

TABLE I summarizes the properties of target systems.

III. Testbed Environment

We utilized *StarBED*[3] [21] which is operated by National Institute of Information and Communications Technology Japan[4]. StarBED provides more than 1000 server nodes interconnected via 1Gbps switches which can be reconfigured to topologies suitable for our experiment plans. More information is available from the web page.

In this experiment, we used 89 server nodes to build distributed file/storage system infrastructure. The server specification is shown in TABLE II.

The upstream switch of the servers were Brocade MLXe-32.

TABLE II
The specification of server nodes

Model	Cisco UCS C200 M2
CPUs	Intel(R) Xeon(R) CPU X5670 @ 2.93GHz × 2
Cores	12 (24 when Hyper-threading is on)
Memory	48GB
HDD	SATA 500G × 2
NIC 0, 1, 2, 3	Broadcom BCM5709 Gigabit Ether
NIC 4, 5	Intel 82576 Gigabit Ether

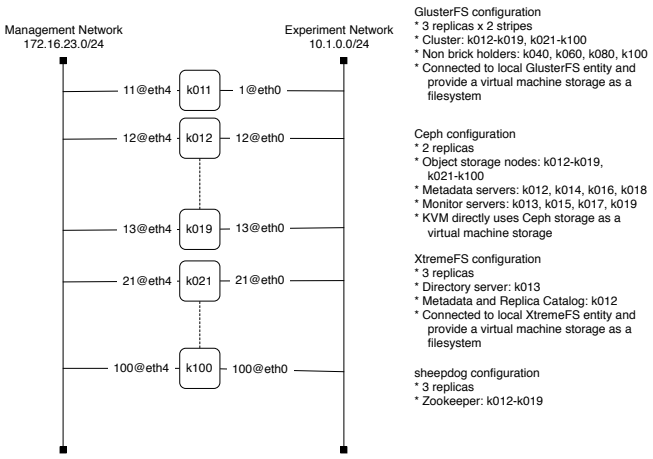


Fig. 1. Node layout of distributed file/storage systems

IV. Layout of Service Entities

A. Measurement Management

Fig. 1 shows the layout of service entities of each distributed file/storage system. The measurement topology consisted of 89 nodes.

The k011 node was used as a controller node from where we sent measurement operation commands. k011 was also used as a NFS server to provide shared space to record measurement result taken on each server node. All the traffic to control measurement scenarios and all the NFS traffic were exchanged using the *Management Network* (the left hand side network of Fig. 1) to avoid any affect to the measurement.

B. Distributed File/Storage Systems

All the nodes were attached to the *Experiment Network* using a BCM5709 network interface port (the right hand side network of Fig. 1). The file/storage systems consisted of 88 nodes, using from k012 to k100 except k020. Most of the

[1] http://ceph.com/wiki/QEMU-RBD/
[2] http://github.com/collie/sheepdog/wiki/Getting-Started
[3] http://www.starbed.org/
[4] http://www.nict.go.jp/en/index.html

TABLE I
SUMMARY OF TARGET FILE/STORAGE SYSTEMS

	Ceph	Sheepdog	GlusterFS	XtreemFS
Metadata management	Multiple metadata servers	n/a	Distributed	Centralized
Data management	CRUSH	Consistent hasing	Consistent hashing	Key-Value
Storage style	Object-based	Object-based	File-based	Object-based
File system	Yes	No	Yes	Yes
QEMU native support	rbd	sheepdog	No	No
Data access entry point	Monitor node	any Sheepdog node	any GlusterFS server node	Metadata server
WAN awareness	No	No	No	Yes

nodes worked as a simple storage node entity, except a few nodes which had some special roles as described below.

- Ceph
  Ceph requires two kinds of special nodes for its operation. One is a *Metadata Server*, the other is a *Monitor*. In the measurement topology, k012, k014, k016, and k018 acted as metadata servers. k013, k015, k017, k019 acted as monitors. These nodes also acted as storage devices (it is called as *Object Storage Node, OSD*). The rest of the nodes acted as OSDs.
  Ceph keeps multiple copies of data object for redundancy. The number of the copies kept in the system is configurable. In the experiment, we chose 2 as the replication factor in the system.
- Sheepdog
  Sheepdog requires a group communication infrastructure to monitor nodes joining and leaving the storage entity. In the experiment, Apache ZooKeeper was used for that purpose. k012 to k019 were used for the ZooKeeper service. All the nodes including k012 to k019 acted as participating nodes of Sheepdog.
  The Sheepdog storage system was formatted to use a replication factor of 3 in this experiment.
- GlusterFS
  The metadata management in GlusterFS is fully distributed, so there is no special nodes like a metadata server. GlusterFS provides the replication and striping function for redundancy and performance enhancement. In the experiment, the replication factor was set to 3 and the striping factor was set to 2.
  Since GlusterFS requires the total number of nodes to be a multiple of the value of replicas $\times$ stripes, the number of participating nodes must be divisible by 6 in this case. Therefore, k040, k060, k080, and k100 were not used as a participating node in GlusterFS. The remaining 84 nodes were used to construct the GlusterFS system.
- XtreemFS
  XtreemFS requires two kinds of special nodes. One is a *Directory Server*, the other is a *Metadata and Replica Catalog, MRC*. k013 was used as a directory server, and k012 was used as a MRC.
  As similar to other systems, XtreemFS can have multiple copies of data object. In the experiment, 3 copies of data were kept in the system.

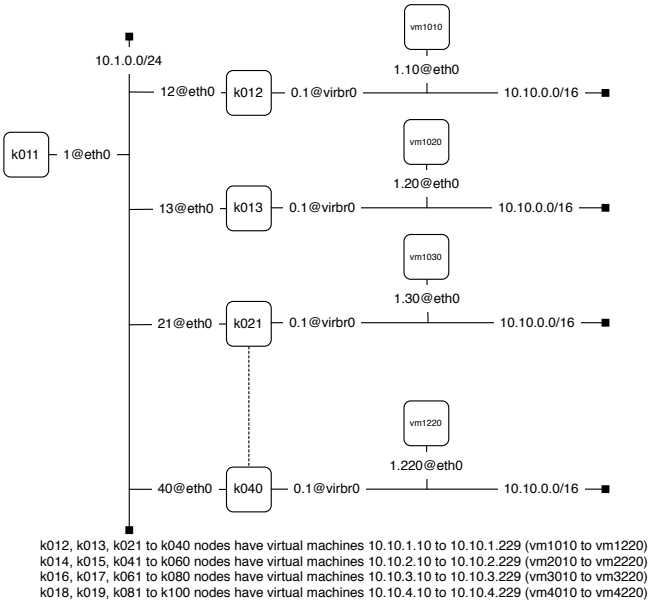


Fig. 2. The layout of virtual machines

Since each server node has two disks, the storage space in each storage nodes was allocated in a different disk (in the second disk) from the boot drive (the first disk) which contains OS and other system software.

C. Virtual Machines

All the nodes except k011 acted as QEMU hypervisors. Fig. 2 shows the topology for virtual machines. Each hypervisor has one virtual machine. The network provided at each hypervisor for virtual machines is a NATed network built by a network bridging function and NAT software. We used the *bridge-utils* package and the *dnsmasq-base* package provided for the Ubuntu 12.04.1 LTS system.

Virtual machine disk images were served by the QEMU hypervisor. Depending on the underlying distributed file/storage systems, the following configuration parameters were used.

- Ceph
  QEMU has a built-in Ceph OSD disk image support as the rbd method. A virtual machine disk image was divided into units of the Ceph OSD object, and distributed through the OSD system.
- Sheepdog
  QEMU has a built-in Sheepdog disk image support as

the sheepdog method. Similar to Ceph, a virtual machine disk image was divided into units of the Sheepdog object and distributed among the storage entities.

- GlusterFS
  At this time of writing, QEMU does not have any direct support of GlusterFS. Since GlusterFS provides an ordinal filesystem interface, we simply mounted the GlusterFS using *FUSE* [22] on each hypervisor and put all the virtual machine disk images into the mounted directory. Each disk image was divided into two pieces (since we used the striping factor of 2), and 3 copies of each piece were distributed among the storage entities.
- XtreemFS
  There is no XtreemFS direct support in QEMU either. Similar to the GlusterFS operation, we mounted the XtreemFS volume on each hypervisor using FUSE and put all the virtual machine disk images in the directory. Each disk image was divided into units of the XtreemFS object and distributed to the storage entities.

V. Measurement Procedures

The main goal of this measurement is to evaluate the maximum performance we can achieve when we use distributed file/storage systems as virtual machine disk image storage systems, and to evaluate their robustness against the number of running virtual machines and network latency. The measurement tool used in the experiment was the *Bonnie++* benchmark software [23].

To evaluate the impact of the scale of the virtual machine infrastructure, we conducted four types of tests:

- Parallel-1
  In the Parallel-1 case, only one virtual machine executes the measurement program. That means the virtual machine can occupy the entire distributed file/storage systems in this case. We run the measurement program on ten different virtual machines sequentially.
- Parallel-10
  In the Parallel-10 case, 10 different virtual machines run the measurement command simultaneously. Each virtual machine is located in different hypervisors.
- Parallel-20
  Same as the Parallel-10 case, except the number of virtual machines is 20.
- Parallel-44
  Same as the Parallel-10 case, except the number of virtual machines is 44.

We also performed the same measurement in a network environment with different latency aiming to emulate wide area network as shown in Fig. 3. We divided 88 storage nodes into 4 groups, each of which has 22 nodes. Nodes in the same group can communicate without any delay, however, nodes in different group have 10ms delay in each direction. This configuration roughly emulates four datacenters located in four different locations.

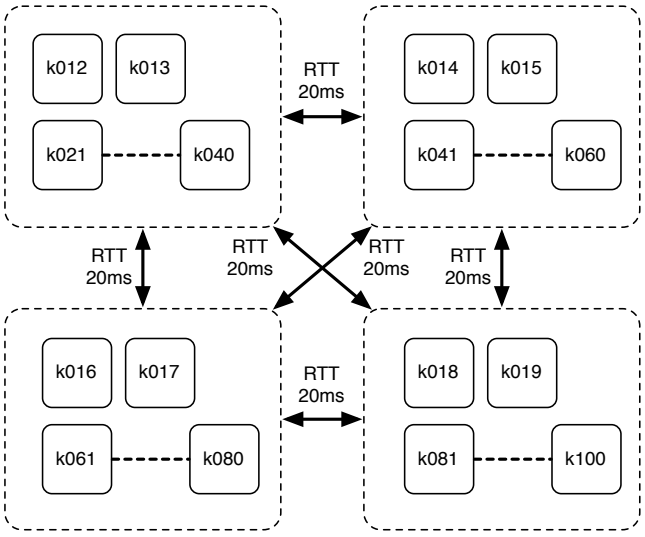


Fig. 3. The latency parameter configuration

VI. Results

A. Write Throughput

Fig. 4 shows the result of the throughput measurement of write operations. Each bar in the figure corresponds to each virtual machine.

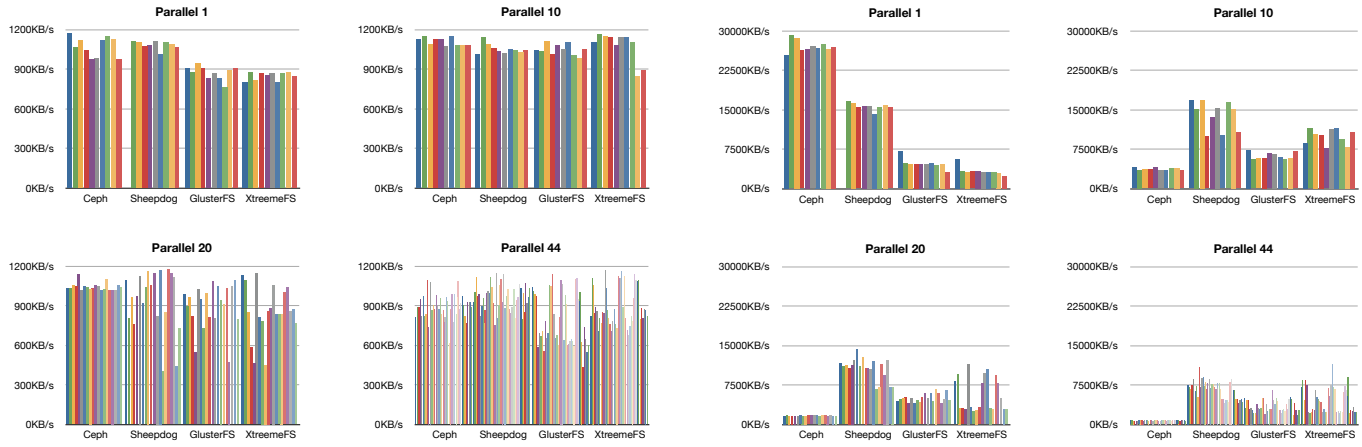
In character write operations (Fig. 4(a)), we can observe Ceph and Sheepdog show slightly better performance than GlusterFS and XtreemFS in the Parallel-1, Parallel-20, and Parallel-44 cases. When the number of parallel operation increases, fluctuation of the throughput is observed especially in Sheepdog, GlusterFS, and XtreemFS cases. Ceph is more stable than others.

In block write operations (Fig. 4(b)), Ceph shows twice as much throughput as that of Sheepdog and five times as much throughput as those of GlusterFS and XtreemFS cases, as long as the number of running virtual machine is one. When we run multiple virtual machines in parallel, Sheepdog gives better performance than others. Sheepdog, GlusterFS, and XtreemFS show some fluctuation in their throughput results. Ceph works more stable than others, however, its throughput is worse especially when the number of parallel virtual machines increases.

B. Read Throughput

Fig. 5 shows the result of the throughput measurement of read operations.

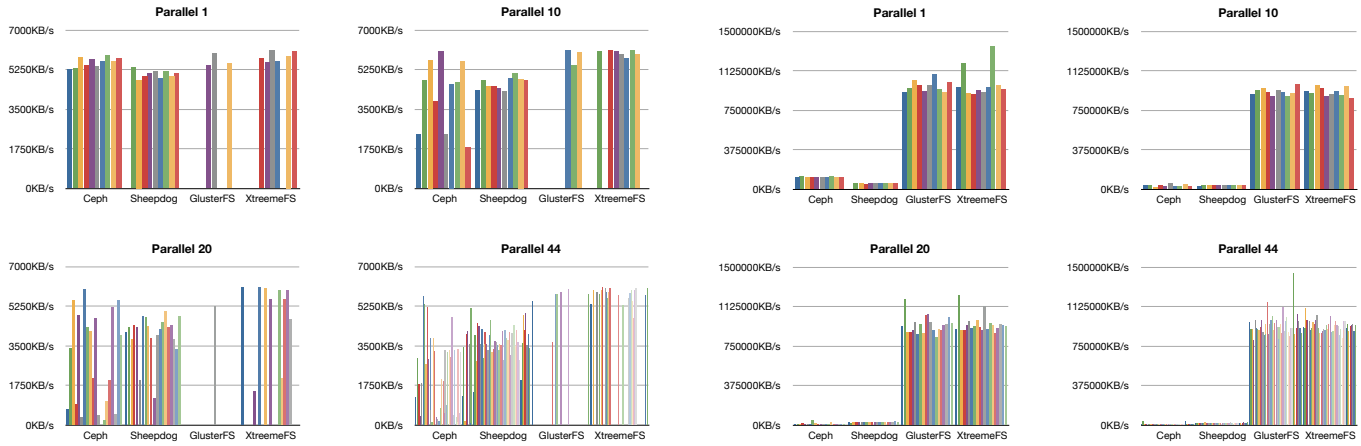
In character read operations (Fig. 5(a)), GlusterFS and XtreemFS perform slightly better than Ceph and Sheepdog. We can observe many missing bars in the result of GlusterFS and XtreemFS. This is expected behavior of Bonnie++. When a test operation completes in too short time, Bonnie++ does not provide any result because it may give inaccurate result. For precise measurement, we need to increase the amount of read operation in the measurement procedure so that we can



(a) Character write

(b) Block write

Fig. 4. Write throughput



(a) Character read

(b) Block read

Fig. 5. Read throughput

get reasonable operation time to calculate trustable value of read throughput.

In block read operations (Fig. 5(b)), GlusterFS and XtreemFS give almost similar performance and they are quite better than Ceph and Sheepdog. In the Parallel-1 case, GlusterFS performs around 8 times and 18 times better than Ceph and Sheepdog respectively. In the Parallel-44 case, GlusterFS performs around 165 times and 48 times better than Ceph and Sheepdog respectively.

We can also observe the number of running virtual machines in parallel affects the performance in Ceph and Sheepdog cases. GlusterFS and XtreemFS seem to be more robust in parallel operations.

C. Throughput with Latency

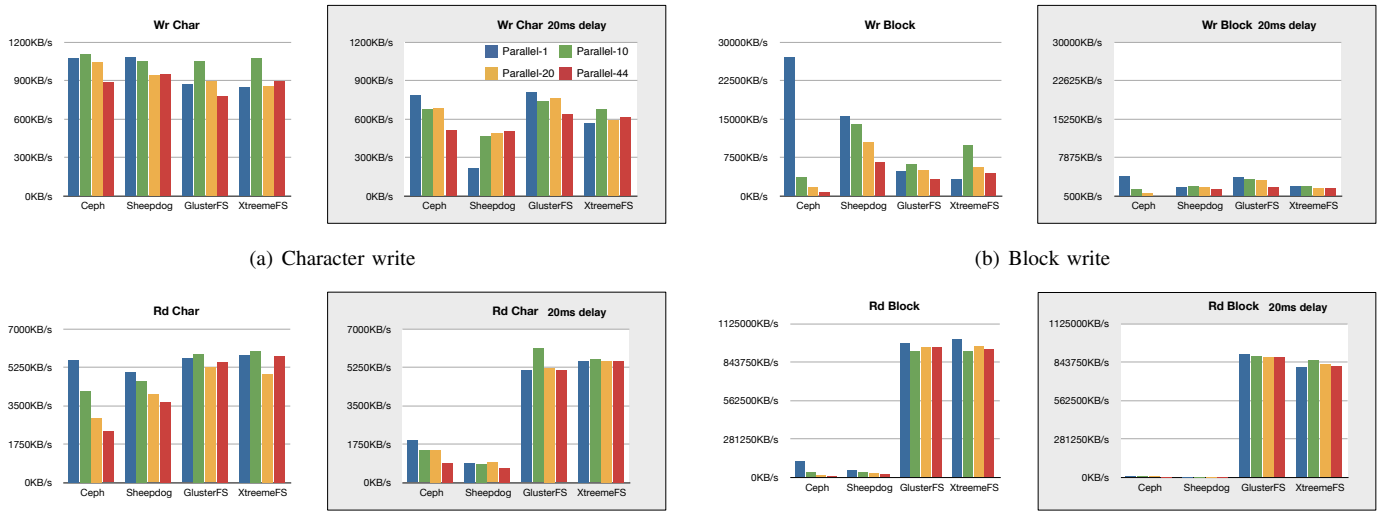
Fig. 6 shows the impact of network latency. In each chart in Fig. 6, the left hand side chart shows the result with no latency, and the right hand side chart (with the gray background) shows the result with 20ms latency as described in section V.

Different from the previous figures, the bars in Fig. 6 show the average throughput of multiple virtual machines. The blue, green, yellow, and red bars indicate the average throughput of the Parallel-1, Parallel-10, Parallel-20, and Parallel-44 cases.

We can observe that in both read and write operation cases, network latency affects the performance of the operations significantly, except for the read operations of GlusterFS and XtreemFS. Their read throughput did not show big degradation even though there was 20ms network latency.

D. Scalability

We can also observe that scalability of GlusterFS and XtreemFS is better than that of Ceph and Sheepdog from Fig. 6. When the number of virtual machines running in parallel increases, Ceph and Sheepdog tend to show performance degradation, while GlusterFS and XtreemFS can keep almost same performance regardless of the number of virtual machines.



(a) Character write

(b) Block write

(c) Character read

(d) Block read

Fig. 6. Impact of latency and impact of the number of virtual machines

VII. Discussion

We observed Ceph and Sheepdog provided good performance in character write operations. Although the difference was not so big actually. The performance of Ceph was around 122% of that of GlusterFS. Sheepdog was almost the same.

Interestingly, the performance of block write operations of Ceph was better than others when we ran only one virtual machine in the entire system, however, when we ran multiple virtual machines, the performance went worst. This result means that the concurrent access to Ceph OSDs have some bottleneck. Technically, the mechanism to provide virtual disk in Ceph is similar to that of Sheepdog. We were initially expecting Ceph and Sheepdog would show similar performance trend, but they were different. In this measurement experiment, we used the rbd method provided by QEMU to serve virtual disks to virtual machines. That method is simply using Ceph OSDs as distributed object stores, and is not using the filesystem part of Ceph. If we used Ceph as a filesystem as similar to GlusterFS and XtreemFS, the result might be different.

For read operations, GlusterFS and XtreemFS provided better performance in both character read and block read operations. Especially in block read operations of the Parallel-44 case, GlusterFS was 164 times better than Ceph, and 48 times better than Sheepdog. XtreemFS performed similarly. The difference between these two groups was that we used QEMU direct support functions for Ceph and Sheepdog (the rbd method as described before, and the sheepdog method respectively) to provide virtual disks to virtual machines, while we used GlusterFS and XtreemFS through hypervisor's filesystem. We think the caching strategy of hypervisor's filesystem contributed the better read performance. As noted before, Ceph can also be used as a filesystem. The performance measurement when we use Ceph as a filesystem is one of our future works.

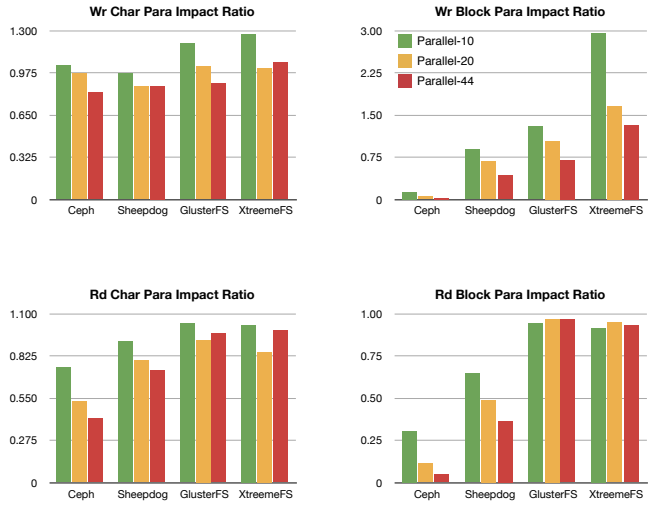


Fig. 7. Parallelism impact ratio

Since we are targeting a wide are operation of virtual machines for geographically distributed datacenters, the impact of latency, and the impact of the number of virtual machines in operation in the system are important evaluation factors.

Fig. 7 shows the parallelism impact ratio calculated from equation 1.

$$ParallelismImpactRatio = \frac{Throughput_{Parallel-X}}{Throughput_{Parallel-1}} \quad (1)$$

Where $Throughput_{Parallel-1}$ is average throughput of I/O operations when we run only one virtual machine at one time. $Throughput_{Parallel-X}$ is average throughput of I/O operations of the parallel case $Parallel-X(X = 10, 20, or 44)$.

In write operations, we can observe the trend that the performance becomes worse as the number of virtual machines

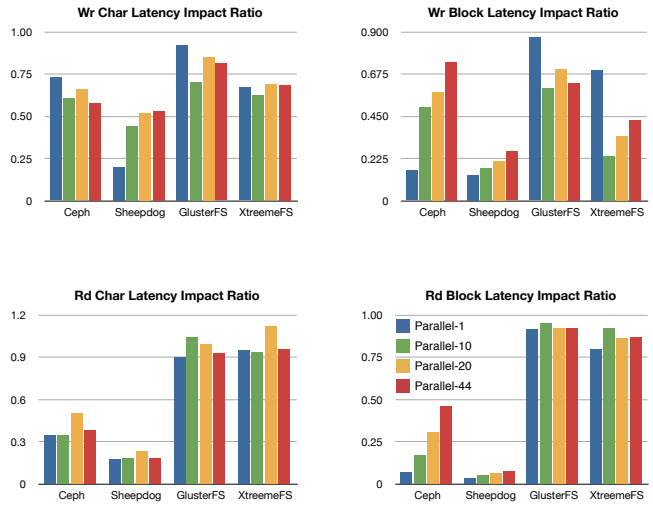


Fig. 8. Latency impact ratio

increases. In read operations, Ceph and Sheepdog have the similar trend, however, GlusterFS and XtreemFS does not have any impact of parallelism.

Fig. 8 shows the latency impact ratio calculated from equation 2.

$$LatencyImpactRatio = \frac{Throughput_{Delay20}}{Throughput_{Delay0}} \quad (2)$$

Where $Throughput_{Delay0}$ is average throughput measured in the base topology without any latency configuration, $Throughput_{Delay20}$ is average throughput measured in the topology with the latency configuration as shown in Fig. 3.

We can observe that GlusterFS is the most robust system against network latency than others. XtreemFS is also good, however, we can see notable throughput degradation in block write operations performed by multiple virtual machines in parallel.

The performance degradation of Sheepdog is most significant when there is a latency in a network.

In block read/write operations of Ceph and Sheepdog, the ratio goes better when the number of parallel virtual machines increases. This means that the impact of network latency becomes smaller and smaller when we have more number of virtual machines. It seems that XtreemFS is also showing similar trend, however, we think we need more data to say so.

For character read/write operations, we cannot see such trend as that of block operations. It seems that the impact of latency is constant in most cases. Only the character read operations of Sheepdog is showing increasing throughput as the number of virtual machine goes up, however, we think we need more data to be confident.

One another important evaluation factor which is not investigated this time is the data transfer efficiency. When we perform a certain amount of read or write operations, we need to measure how much amount of background traffic is required on each distributed file/storage system. The smaller background traffic is better. This is our future task.

VIII. Conclusion

The virtualization technology enabled us to operate more number of computers to host real services. One of the merit of virtualization is that it makes it easy to move virtual resources among physical machines. To achieve more efficient operation of datacenters, we need to provide the mechanism to migrate virtual resources among datacenters to aggregate them to fewer physical machines. To do that, distributed file/storage system which can be operated in a wide area network is required.

In this paper, we picked Ceph, Sheepdog, GlusterFS, and XtreemFS and evaluated them as a backend storage system for virtual machines. The evaluation result shows that GlusterFS and XtreemFS provide far better read performance (we observed 164 times better performance at maximum). Sheepdog provides better block write performance.

We defined two new storage evaluation indexes for a virtualization environment; *ParallelismImpactRatio* and *LatencyImpactRatio*. From these indexes, we observed that the number of virtual machines running in parallel does matter for all the systems. When the number grows, the throughput goes down, however, for block read operations, GlusterFS and XtreemFS are not affected by this factor. The latency impact exists, however, we observed the trend that when we increase the number of virtual machines, the impact is going to small.

From what we have achieved from the experiment, our current suggestion of the distributed file/storage system for virtual machine image storage is GlusterFS or XtreemFS. We still need to evaluate more factors, such as data transfer efficiency, which is important especially in a wide area operation. We will keep investigating the distributed file/storage systems to build better wide area virtualization environment.

Acknowledgments

We would like to thank Toshiyuki Miyachi and Hiroshi Nakai for their support at StarBED and many useful suggestions and comments. We also thank all the staffs at StarBED for helping our experiment.

References

[1] E. Harney, S. Goasguen, J. Martin, M. Murphy, and M. Westall, "The efficacy of live virtual migrations over the internet," in *Proceedings of the 2nd international workshop on Virtualization technology in distributed computing (VTDC'07)*, 2007.

[2] Cisco Systems, Inc., VMware, Inc., "Virtual Machine Mobility with Vmware VMotion and Cisco Data Center Interconnect Technologies," Cisco Systems, Inc., VMware, Inc., Tech. Rep., 2009.

[3] T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A Live Storage Migration Mechanism over WAN for Relocatable Virtual Machine Services on Clouds," in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'09)*, 2009, pp. 460–465.

[4] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn, "Ceph: A Scalable, High-Performance Distributed File System," in *Proceedings of the 7th symposium on Operating systems design and implementation (OSDI'06)*. USENIX, 2006, pp. 307–320.

[5] K. Morita, "Sheepdog: Distributed Storage System for QEMU/KVM," Linux.conf.au 2010, January 2010.

[6] Gluster Inc., "Gluster File System Architecture," Gluster Inc., Tech. Rep., 2010.
[7] E. Cesario, T. Cortes, E. Focht, M. Hess, F. Hupfeld, B. Kolbeck, J. Malo, J. Martí, and J. Stender, "The XtreemFS Architecture," in *Linux Tag*, 2007.
[8] F. Bellard, "QEMU, a Fast and Portable Dynamic Translator," in *Proceedings of the annual conference on USENIX Annual Technical Conference (ATEC'05)*, April 2005, pp. 41–41.
[9] ——, "QEMU: Open Source Processor Emulator," http://www.qemu.org/.
[10] R. Harper, A. Aliguori, and M. Day, "KVM: The Linux Virtual Machine Monitor," in *Proceedings of the Linux Symposium*, 2007, pp. 225–230.
[11] ——, "KVM: Kernel-based Virtual Machine," http://www.linux-kvm.org/.
[12] Inktank Storage, Inc., "Ceph," http://ceph.com/.
[13] S. A. Weil, A. W. Leung, S. A. Brandt, and C. Maltzahn, "RADOS: A Scalable, Reliable Storage Service for Petabyte-scale Storage Clusters," in *Proceedings of the 2th international Petascale Data Storage Workshop (PDSW'07)*, November 2007, pp. 35–44.
[14] S. A. Weil, S. A. Brandt, E. L. Miller, and C. Maltzahn, "CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data," in *Proceedings of the 2006 ACM/IEEE conference on Supercomputing (SC'06)*, November 2006.
[15] R. Honicky and E. L. Miller, "Replication under scalable hashing: A family of algorithm for scalable decentralized data distribution," in *Proceedings of the 18th International Parallel & Distributed Processing Symposium (IPDPS 2004)*, April 2004.
[16] K. Morita, "Sheepdog," http://www.osrg.net/sheepdog/.
[17] S. Dake *et al.*, "Corosync," http://www.corosync.org/.
[18] The Apache Software Foundation, "Apache ZooKeeper," http://zookeeper.apache.org/.
[19] Gluster Inc., "Gluster," http://www.gluster.org/.
[20] The Contrail E.U. project, The MoSGrid project, and The First We Take Berlin, "XtreemFS," http://www.xtreemfs.org/.
[21] T. Miyachi, K. Chinen, and Y. Shinoda, "StarBED and SpringOS: large-scale general purpose network testbed and supporting software," in *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools (valuetools'06)*, October 2006.
[22] M. Szeredi, "FUSE: Filesystem in Userspace," http://fuse.sourceforge.net/.
[23] R. Coker, "Bonie++," http://www.coker.com.au/bonnie++/.