Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186–5140
Volume 3, Number 1, pages 1–7, January 2014

Towards Fast Deploying the Coordinated Storage, Computing, and Networking Cloud Service: the Ezilla Toolkit

Chi-Ming Chen, Yi-Lun Pan, Hsi-En Yu, Chang-Hsing Wu, Hui-Shan Chen, and Kuo-Yang Cheng
National Center for High-Performance Computing
Hsinchu, Taiwan
{jonchen, serenapan, yun, hsing, chwhs, kycheng}@nchc.narl.org.tw

Abstract—With the growth of Cloud based technologies, so have the cloud-based services has grown over the past few years. To meet the demands from the Cloud users, service providers have to provide such a service environment that can be deployed and provisioned quickly and easily. To help tackling such an issue, the Ezilla, which is considered as a private Cloud toolkit, has been developed by the PerComp Team in the National Center for High-performance Computing. The Ezilla integrates the de facto Cloud middleware, Web-based interface, and coordinated cloud infrastructure services such as Storage, computing and networking services to form an integrated virtual computer. With the Ezilla toolkit, a virtual computer, which is configured specifically to meet the needs of an individual Cloud user, is just one click away. The merit of the Ezilla is simplifying the complexity of the Cloud system for easier deployment for the users. Scientist as well as genera users can deploy the Cloud for their works without dealing with the painful process of building the Cloud system. The proposed Ezilla toolkit leverages genetic virtualization techniques, DRBL–SSI (Diskless Remote Boot in Linux – Single System Image), the cluster scheduling policy, and distributed filesystem, to orchestrate distributed computing resources dynamically.

Keywords—Cloud; Virtualization Techniques; Virtual Cluster, DRBL;Scheduling Policy; Distributed Filesystem;

I. INTRODUCTION

In Cloud computing environment, there are various important issues, including information security, virtual computing resource management, and so on. Among these issues, the virtual computing resource management has emerged as one of the most important issues in the past few years. Currently, Cloud users have to manually build specified virtual cluster using commends in order to generate and manage virtual resources. To improve the situation, the Ezilla toolkit [1] has been developed by the PerComp Team of the National Center for High-Performance Computing (NCHC). Technology-wise, the Ezilla leverages unattended installation technique, cloud middleware, clone OS toolkit, and fast deployment toolkit, which is Diskless Remote Boot in Linux - Single System Image (DRBL-SSI) [2], to provide the software infrastructure of the Ezilla system. Furthermore, in order to provide Cloud users with an environment in a friendly and straightforward manner, a user interface, the Ezilla Web Interface, is introduced to offer a seamless and unified access to geographically distributed resources connected via the Internet. The Ezilla toolkit helps Cloud users to build and manage their own private cloud as easy as one single click, thus to lower the barrier of using the Cloud computing environment. This research focuses on virtual resources management with an interactive graphical user environment.

An efficient scheduling policy and distributed filesystem are both indispensable, especially for distributed computing and Cloud computing. Thus, the distributed filesystem is adopted and also built-in Ezilla, which includes GPFS [3] and MooseFS [4].

The ultimate goal of this research is to find a solution for scientists/researchers to run their jobs on Clouds without having to deal with the complexity of the Cloud technologies. This research focuses on the development of a friendly user interface, automatically dynamic resource allocation technique and integrated heterogeneous computing resources. The rest of the paper is organized as follows. The Section 2 presents related works. In Sections 3, it shows Ezilla system architecture and Ezilla fast deployment approach. Following discussed by the research results of Ezilla toolkit and experimental performance evaluation presented in the Section 4. Finally, the concluding remarks and future directions are shared in Section 5.

II. RELATED WORKS

A. Virtualization Technologies

The virtualization technology is not a brand new technology. In the late 1990s, virtualization was achieved by complex software (binary translation) techniques given the fact that the virtualization technique was not supported by processors at the moment and obtained reasonable performance. In the late 2006, both the Intel and the AMD created new processor extensions to their processors and enhanced the support of the virtualization. Furthermore, they also implemented the I/O virtualization technology that covers memory, disk and network by the chipset. The technology was called hardware-assisted virtualization. This approach increases the performance by removing a layer of complex software techniques between the guest OS and hardware. Examples of virtualization platforms that are adapted to such hardware include Linux KVM, VMware Workstation, Microsoft Hyper-V, to name a few.

In the arena of virtualization technology, hypervisor is considered as a crucial part of the technology. It is software that manages multiple virtual machines on a single computer system. Hypervisor is a layer resides between the existing operating system and hardware to manage the computing hardware and virtual machines. The characteristics of virtualization technology are described as the following:

Utilization – better utilization means various services run on one physical machine with multiple virtual machines (VMs);

Isolation - better isolation means a VM can halt and catch fire without affecting the real host or other running VMs;

Flexibility - the ability of the virtualization technologies to run platforms and operating systems that are different from the host, good flexibility means more choices for VM platforms and the ability to run VMs with minimal modification;

Manageability - availability of tools and APIs for starting, stopping and moving VMs.

Generally, modern hypervisor implementations are divided into two categories, including Host-based and Bare-metal approaches. The host-based approach uses modified operating systems to provide virtual machine monitoring, such as Linux-VServer [5], Solaris Zones [6], and Kernel-based Virtual Machine (KVM) [7]. On the other hand, the bare-metal approach employs small-dedicated hypervisors to directly run on physical machines. The VMware ESX server [8], and the XenServer [9] are the famous examples of the bare-metal approach.

With success of the virtual technologies, we integrate virtualization technology – KVM and Web platform. This research comes up with a new and lightweight approach to access virtual computing services via the Ezilla toolkit.

B. *Cloud Middleware*

OpenNebula [10] is one of the most advanced cloud computing platforms in the open source community. Project tends to be customizable to meet the demand from the adopter of the software. The OpenNebula toolkit manages a data center's virtual infrastructure to build private, public and hybrid IaaS (Infrastructure as a Service). It is used as central cloud management and responsible for allocating available computing resources, creating VMs based on selected VM image, and deploying the chosen image onto the physical computing resources for service. However, the OpenNebula lacks the automatic installation and configuration mechanism. Therefore, users have to install and configure OpenNebula manually.

In addition to OpenNebula, the OpenStack [11] is a collaborative software project designed to create freely available code and needed standards for the benefit of both Cloud providers and Cloud customers. OpenStack consists of three components, as shown in the following:

OpenStack Compute – It can deploy automatically provisioned virtual compute instances;

OpenStack Object Storage – It is responsible for redundant storage of static objects;

OpenStack Image Service – It provides discovery, registration, and delivery services for virtual disk images.

The functions of OpenStack are complete; however, the installations and configurations of OpenStack are still too complex for general users to handle. To solve above issues, the Ezilla toolkit takes the advantage of the kickstart [12] mechanism used in Linux installation. The kickstart is a component (i.e. command-line application) wrapper that provides a range of information related to component execution including exit status, signals, resource usage, and compute environment descriptions. The Ezilla server leverages the kickstart mechanism of fast deployment and the lighter cloud platform, which is OpenNebula currently, for managing heterogeneous distributed data center infrastructures. The complete installation and configuration procedure can be finished automatically and easily by the Ezilla toolkit

III. THE IMPLEMENTATIONS OF THE EZILLA TOOLKIT

A. *Research Objective*

The key idea of Cloud Computing lies in its component-based nature, which are reusability, substitutability and user friendly. By integrating virtualization technologies, DRBL – SSI mode, distributed filesystem, cluster scheduling policy module, and a web environment to access Cloud services via Ezilla Web Interface is provided. This progress helps to lower barrier of using Clouds. More specifically, the Ezilla toolkit implements an autonomic virtual computing resources management system based on decentralized resource discovery architecture. At the same time, an efficient scheduling policy is also crucial to the utilization of Cloud resource. Therefore, the designed cluster scheduling policy module is used for the scheduling of virtual cluster and physical cluster. It can orchestrate the computing resources to provide the ultimate resource allocations.

B. *Implementation and SystemArchitecture*

There are three components in the Ezilla toolkit, including Ezilla Server, Ezilla Client, and intuitive Ezilla Web Interface, as sketched in the Fig. 1. The Ezilla Server is resporble for orchestrating computing resources, and monitoring the status of physical and virtual machines via cluster scheduling module. In addition, the most important feature is the image file server used to manage virtual machine images. The VM images are stored in the file system located in the Ezilla server. Responding to the request from Cloud users, the images will be dispatched to Ezilla clients and used to generate virtual machines in turn by Ezilla clients. Once the Ezilla Server is in place, Ezilla Clients can be setup automatically via the DRBL–SSI and existing virtualization technologies. With such a mechanism implemented, a physical machine can be easily turned into Ezilla Client via PXE with the local disk untouched. Hence, computing resources can be added dynamically without any reconfiguration, thus to enhance the flexibility of Cloud resource allocation. A Cloud user can use his personal Ezilla Web interface to create a Cloud environment integrating storage, networking, and computing power timely and easily.

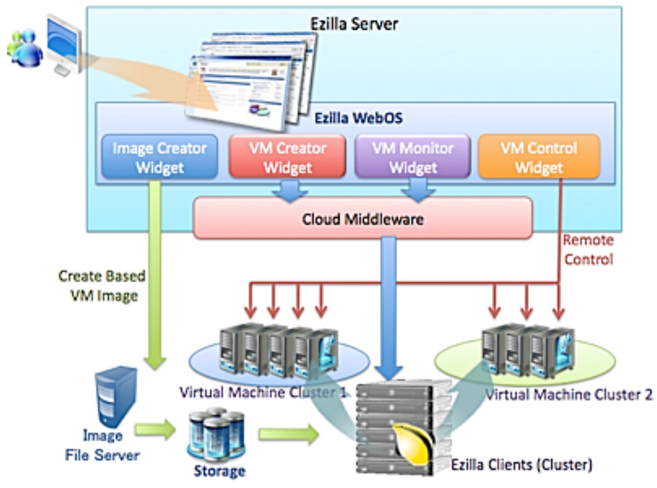


Fig. 1. The System Architecture of Ezilla

C. *Ezilla Fast Deployment Approach*

In the Fig. 2, it is the scenario of Ezilla fast deployment approach. At first, Cloud users just boot the Ezilla Server with a genuine Debian or Ubuntu CD, and then detect the network card to connect Internet. After that, the process of unattended installation is starting automatically until the whole installation procedure is completed. Due to DRBL-SSI, a physical machine can be easily turned into Ezilla Slave via PXE with the local disk untouched via Ezilla diskless approach. Hence, computing resources can be added dynamically without any reconfiguration, thus to enhance the flexibility of Cloud resource allocation.

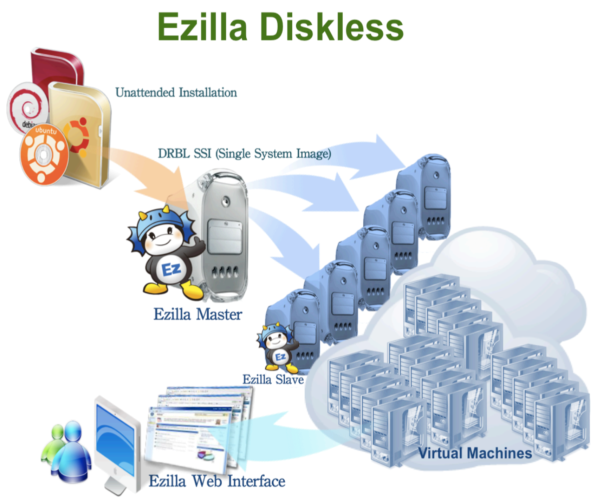


Fig. 2. The logical procedure of Ezilla Fast Deployment Approach

Each step of Ezilla fast deployment approach is described as the following pseudo code in the Fig. 3.

This kind of installation was designed to be as effortless as possible so that a Cloud environment can be deployed quickly and easily. In fact, an Ezilla Server can be deployed in three simple steps via DRBL-SSI mode.

Ezilla Master installation

Input: *Using Ubuntu's* **preseeding** *for automating Ezilla Master installation*

```
1. Select first disk to partition and formation
2. Add DRBL repository
3. Install Linux OS into disk
4. Install needed packages for virtualization, such as KVM,
libvirt, ruby and so on.
5. Copy OpenNebula package, which modified by Percomp
laboratory and Ezilla initiation script into Ezilla Master
6. run Ezilla initiation script
   for Ezilla slave
   - Add support of KVM physical network sharing bridge
   - mount share storage for vm image
   - modify libvirtd.conf
-change unix_sock_group and unix_sock_rw_perm
   - modify qemu.conf
   -change user/group and dynamic_ownership
7. Create necessarily users and generate users' ssh key
8. Add service for DRBL initiation during the first time
booting in Eziila Master

The first time booting in Eziila Master
1. Runing DRBL initiation script
  - install needed packages for DRBL, such as tftp, dhcpd,
nfs and so on.
  - copy needed system files to /tftproot
  - generate Ezilla Slave's system image for DRBL SSI
mode
  - add needed service for DRBL
2. Turn on Ezilla slave's power and enjoy it!!
```

Fig. 3. The Pseudo Code of Ezilla Fast Deployment Approach

- Step1 – Boot the server with a genuine Debian or Ubuntu CD and select “Ezilla Cluster”. As the Fig. 4 is shown.

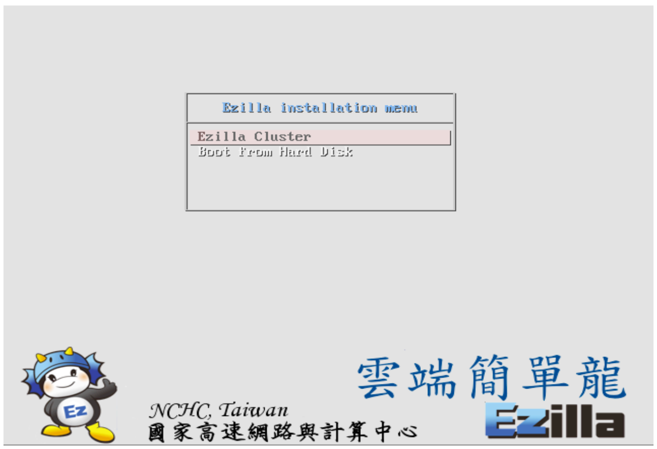
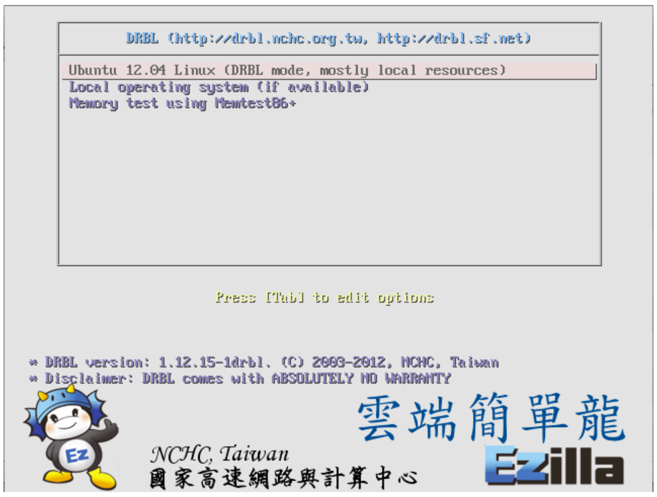


Fig. 4. Install the Ezilla Master

- Step2 – Use DHCP or configure the IP address to connect the Internet.
- Step3 – The unattended installation is starting automatically, until the whole precedure is finished.

After completion of all the three steps above when the Linux server is ready for further use, the automatic installation process for building the Ezilla environment takes over. The installation will proceed without requiring the attention from a system administrator. The installation is completed and the Ezilla Cloud environment is ready for use once the boot menu page is shown, as in the Fig. 5



```
DRBL (http://drbl.nchc.org.tw, http://drbl.sf.net)

Ubuntu 12.04 Linux (DRBL mode, mostly local resources)
Local operating system (if available)
Memory test using Memtest86+

Press [Tab] to edit options

* DRBL version: 1.12.15-1drbl. (C) 2003-2012, NCHC, Taiwan
* Disclaimer: DRBL comes with ABSOLUTELY NO WARRANTY
```

Fig. 5. The Boot Menu on a Ezilla Slave after the Installation

IV. RESULTS AND THE PERFORMANCE EVALUATION

A. *Ezilla Web Interface*

After the automatic installation of Ezilla platform, we also delegate the friendly Ezilla Web interface. Via web browsers, the Cloud users can easily manage and control all behaviors of VMs with minimum learning efforts, even if users want to convert physical machine to virtual one directly. There are some functions in Ezilla Web interface, including Dashboard, System, Virtual Resources, and so on. The function of Dashboard – As long as Cloud users log in Ezilla Web interface, they can monitor the status of virtual machines, networks, and the physical hardware. It also shows the current loading of physical machines, as illustrated in the Fig. 6. Furthermore, the purpose of System is to manage privilege of users, groups, and access control lists.

The most important function is Virtual Resource, because it can deal with the configuration of virtual machines, templates, images, and even P2V. The main task of Virtual Resources is with the profile of virtual machine demanded by the user provided, this function generates a specification, as shown in the Fig. 7, which in turn is parsed by the VM generator engine to generate specific virtual cluster on the physical computing resources. In brief, with criteria of the virtual cluster specified by the user, with one click, a virtualized Cloud environment customized for a particular user will be configured in no time. Besides, this function also provides users with the access to VMs through VNC and SSH via web, so users are available for accessing the VMs. Moreover, Cloud users can monitor the detail status of specified VM via web, as the Fig. 8 is shown.

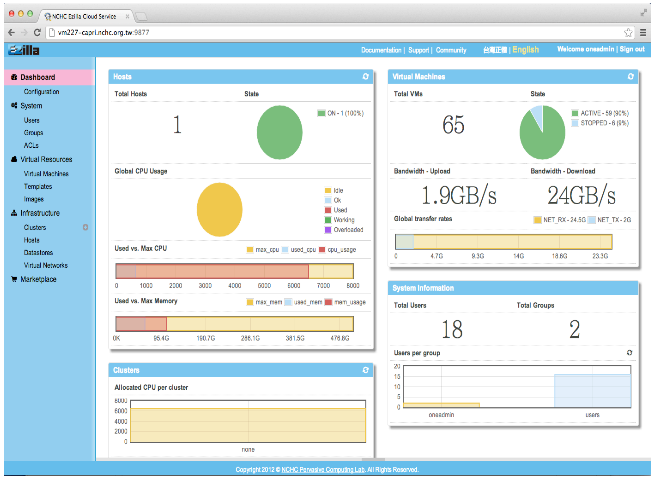


Fig. 6. Ezilla Web Interface

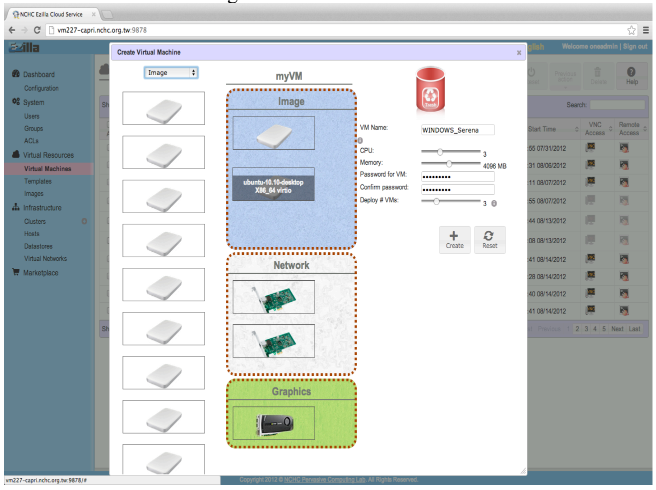
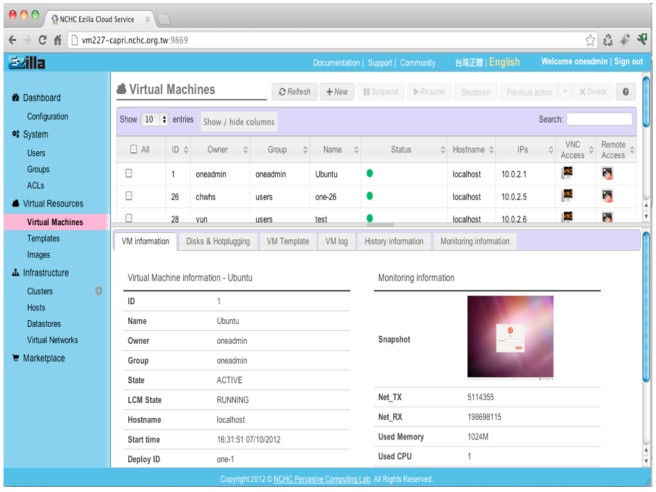


Fig. 7.Generation of VM/Virtual Cluster via “Drag and Drop”



All	ID	Owner	Group	Name	Status	Hostname	IPs	VNC Access	Remote Access
☐	1	oneadmin	oneadmin	Ubuntu	●	localhost	10.0.2.1		
☐	26	chwhs	users	one-26	●	localhost	10.0.2.5		
☐	28	yun	users	test	●	localhost	10.0.2.6		

Virtual Machine information - Ubuntu

ID	1
Name	Ubuntu
Owner	oneadmin
Group	oneadmin
State	ACTIVE
LCM State	RUNNING
Hostname	localhost
Start time	16:31:51 07/10/2012
Deploy ID	one-1

Monitoring information

Net_TX	5114355
Net_RX	198698115
Used Memory	1024M
Used CPU	1

Fig. 8. Monitoring the status of VM/Virtual Cluster via Web

B. *Experimeenta Results*

1) *Experiment Environment*

A series of experiments are performed to evaluate the performance of the system. The environment set up for the experiments includes the following components: the multi-sites physical computing environment, the virtual machine – KVM, and Disk I/O test tool – dd and IOzone [13]. The environment characteristics of the computing resources for the experiments are summarized in the Table I.

TABLE I. SUMMARY OF ENVIRONMENT CHARACTERISTICS OF EXPERIMENTAL COMPUTING RESOURCES

Resource	CPU Model	Memory (GB)	CPU Speed (MHz)	#Cores	Nodes	Network
Capri Cluster	Intel(R) Xeon(R) CPU E5620 , 2.40GHz *2	96GB DDR3 ECC	2400	256	32	10GE *2

The detailed hardware information of the computing resources used for experiments is listed in the Table 1. The Capri Cluster has 32 nodes, and each node has 8 CPU cores. In other words, the total number of cores is 256 cores in Capri Cluster. Therefore, it was used to benchmark the performance of distributed file systems.

2) *Performance Results of Distributed FileSystem*

With distributed file system, it is easier to generate and manage large amount of VMs in real time. The Ezilla toolkit supports three kinds of distributed file system, the GPFS and the MooseFS. The computing – Capri cluster is used to construct these two distributed file systems for experiments to be carried out.

As shown in the Fig. 9 and Fig. 10 respectively, 5 nodes were used to construct the file system for all the two file systems to be compared.

For the environment of GPFS, 5 nodes were used to setup Network Shared Disk (NSD) of GPFS, with one-node serves as the quorum-manager. Similarly, we also use five nodes to be built the chunks of MooseFS (MFS). In the meanwhile, there is one node used as the master node. Thus, the equivalence of the storage system used for experiments is guaranteed.

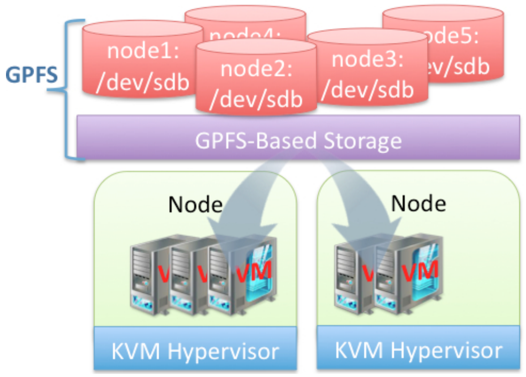


Fig. 9. Alternative of distributed file system to VMs – GPFS

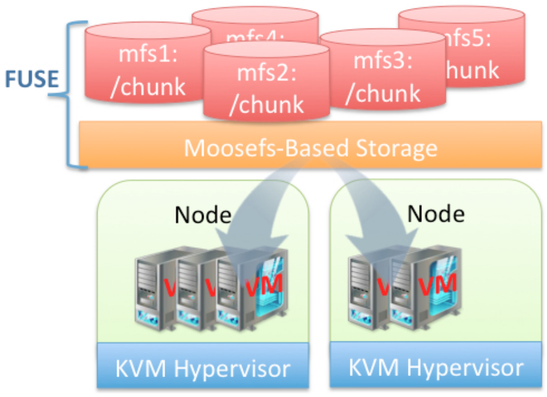


Fig. 10. Alternative of distributed file system to VMs – MooseFS

In the following experiments, the Ezilla was sued to process 10 GB of data via VMs to access both distributed file systems under investigation, which are the GPFS and the MooseFS.

The first experiment was performed using dd, which is a common Unix program with its primary purpose as the low-level copying and conversion of raw data. With the following command line (CL 1), one file, named "test", with 10GB in size was generated on every VM, which mount extra hard disk image (/mnt/disk) from the GPFS and MooseFS distributed file system.

/bin/dd if=/dev/zero of=/mnt/disk/test bs=1024k count=10k (CL1)

Writing the 10 GB "test" file on distributed filesystem carried out this experiment. The total data written to the distributed file system grew along with that of the number of VMs involved. For the case of 1 VM, only 10 GB of data was written to the file systems, while the case of 32-VM wrote 320 GB of data in total to the file systems.

The results, as shown in the Fig. 11, indicate that the maximum I/O access speeds to the GPFS and MooseFS (MFS) were limited to 70 ~ 110 MB per second by using the Virtio [14], in the case of 1-VM. Since the MFS adopts memory as the I/O cache, the I/O speed of the MFS is generally faster than the GPFS. For the specific case of 1-VM, the performance of the MFS exceeds that of the GPFS for about 40MB per second. As the consequence of using I/O cache to boost the performance, the MFS suffers significant performance drop when the number of VMs reach 18 in this case. This is due to the fact that the memory requested by the data reach 180 GB in addition to the memory take by the VMs exceeded the physical memory capacity on the server.

However, the performance was raised when the number of VM reach 21. The phenomenon was due to the activation of the mechanism of parallel write into the parallel file systems. As a consequence, the amount of data queued in the data cache started to reduce that, in turn, help the improvement of system performance.

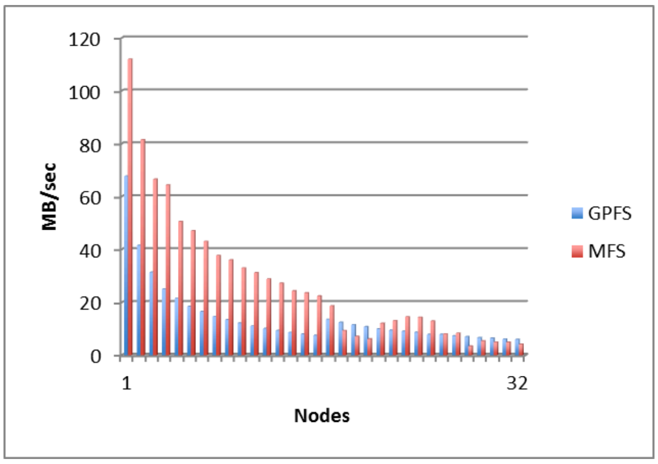


Fig. 11. Comparison between GPFS and MFS using dd to dump 10 GB of data

The IOzone was chosen to perform the second I/O experiment test performance with following command (CL 2).

iozone -s 3g -a -o -r 1m -f /mnt/disk/test -Rb /tmp/iozone.xls (CL 2)

In the CL2, the parameter "-s 3g" indiates that the file size was 3 GB, with the parameter "-r 1m" to define the record size to be 1 MB which provides the best I/O performance for both.

According to the Fig. 12 and Fig. 13, with record size as 4KB, the GPFS provided better I/O-write performance (~ 400 MB/s) than MFS (~ 280 MB/s). However, as the record size grew, the performance of MFS started to catch up with that of GPFS. When the record size reach 1024 KB, the MFS hit the record high of 800MB/s, while the GPFS hit 750 MB/s. Therefore, 1 MB is chosen for benchmarking the performance of distributed file systems.

When the file size kept growing and finally reached 524288 KB (512 GB), the I/O performance deterioration of MFS was less than that of the GPFS (need numbers to back up this statement). Therefore, it is a good choice for the Ezilla to use the MFS as the distributed file system, not only because it is open source solution, but also for its competitive performance w.r.t. the GPFS.

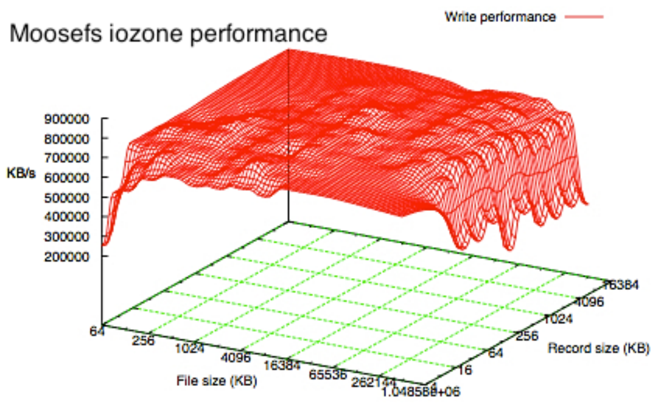


Fig. 12. Writing Rate of Different Record Sizes on MFS using IOzone

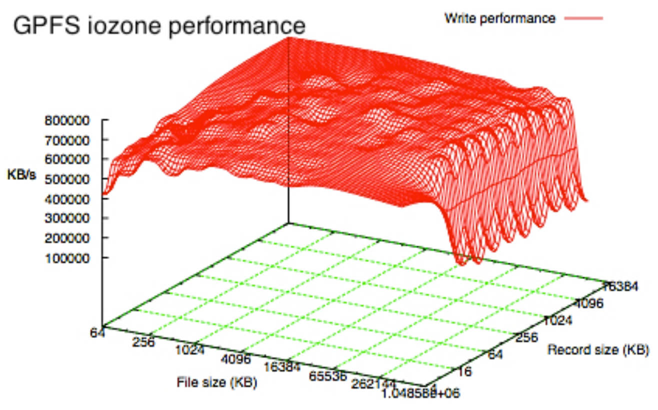


Fig. 13 Writing rate for different record sizes on GPFS using IOzone

Obviously, the MFS show better performance on both Writing/Reading than GPFS, as shown in the Fig. 14 and Fig. 15. The major factor for the MFS to our platform the GPFS was because the memory of the computer in Capri Cluster has 96GB memory. Thus, the memory demanded by I/O cache of the MFS was met during the experiment. As long as the memory is not completely consumed, the MFS has high stability when large data was processed.

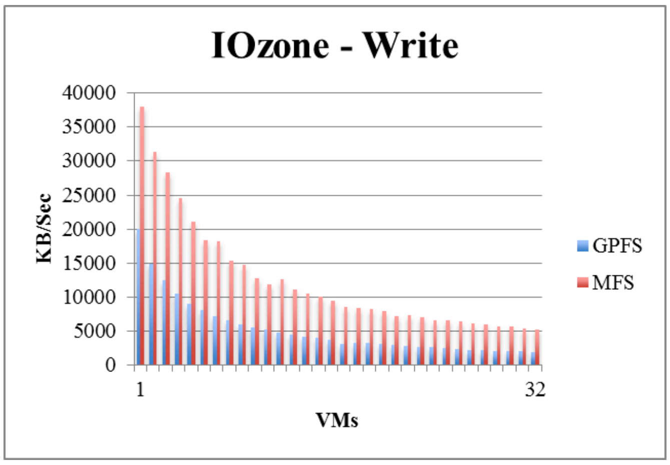


Fig. 14. Comparisons between GPFS and MFS with IOzone Writing

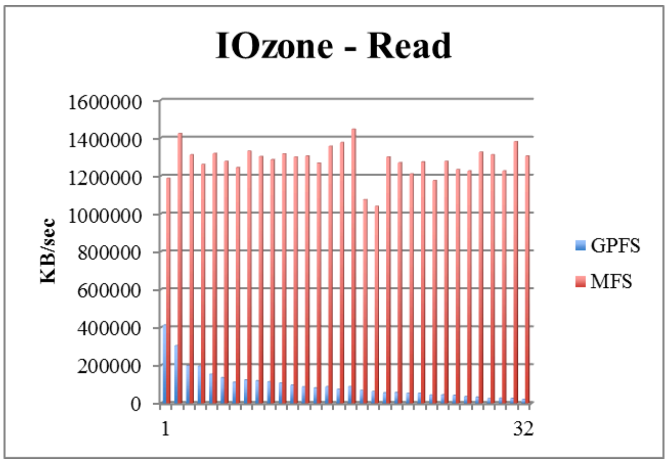


Fig. 15. Comparisons between GPFS and MFS with IOzone Reading

V. CONCLUSION AND FUTURE WORK

The research work – the Ezilla toolkit, provides Cloud users with a friendly, straightforward and yet efficient user interface. The Ezilla integrates the de facto Cloud middleware, and coordinates Cloud infrastructure services (such as storage, computing, and network) to form a virtual computer in the distributed computing environment. With one click, Cloud users can easily customize and configure virtual clusters specified for particular application environment via the Ezilla toolkit. It can not only help user to build the virtual environment easily and automatically, but also provide different varieties of computing environment such as Linux, Windows, and so on. Furthermore, with the DRBL-SSI mode embedded, the complete Ezilla environment can be installed with ease, thus to provide Cloud users with users' own private clouds easily and quickly.

From the results for the distributed file systems, compared to the GPFS, the MooseFS presented itself as a compatible open source alternative for distributed file system. As long as the memory supply does not exhaust, the performance of the MFS can be satisfactory to meet the demand from the Cloud environment. This indicates that the MooseFS is worthy of investigation for the distributed file system under the Cloud environment. The next research will support more powerful distributed filesystem, which is Ceph [15] filesystem.

REFERENCES

[1] Yi-Lun Pan, Chang-Hsing Wu, Hsi-En Yu, Hui-Shan Chen, Weicheng Huang, "Creating Your Own Private Cloud: Ezilla Toolkit - For Coordinated Storage, Computing, and Networking Services," The 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012.

[2] DRBL, http://drbl.org/ , 2013.

[3] Gupta, K., Jain, R., Koltsidas, I., Pucha, H., Sarkar, P., Seaman, M., and Subhraveti, D. , "GPFS-SNC: An enterprise storage framework for virtual-machine clouds," IBM Journal of Research and Development, pp. 2:1-2:10, 2011.

[4] Moose, URL: http://www.moosefs.org, 2013.

[5] S. Soltesz, H. P¨otzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based Operating System Virtualization: A Scalable, High-Performance Alternative to Hypervisors, " Proceedings of ACM SIGOPS/Eurosys European Conf. on Computer Systems, pp. 275-287, Mar. 2007.

[6] D. Price and A. Tucker, "Solaris Zones: Operating Systems Support for Consolidating Commercial Workloads, " Proceedings of 18th Large Installation System Administration Conf., pp. 241-254, Nov. 2004.

[7] B. Zhang, X. Wang, R. Lai, Y. Liang, Z. Wang, Y. Luo, and X. Li, "Evaluating and Optimizing I/O Virtualization in Kernel-based Virtual Machine (KVM)," International Conference on Network and Parallel Computing, pp. 220-231, Zhengzhou, China, September 13-15, 2010.

[8] John Paul, "VMWare ESX Server Workload Analysis: How to Determine Good Candidates for Virtualization," 33rd International Computer Measurement Group Conference, pp. 483-484, San Diego, CA, USA, December 2-7, 2007.

[9] X. Ge, H. Jin; S. Wu, X. Shi, W. Gao, "A method of multi-VM automatic network configuration," Intelligent Computing and Intelligent Systems, pp. 309-313, 2009.

[10] Milojičić D., Llorente I., Montero R., "OpenNebula: A Cloud Management Tool," Internet Computing, IEEE 2011, 15 (2), 11-14, 2011.

[11] Mahjoub M., Mdhaffar A., Halima R., Jmaiel M., "A comparative study of the current Cloud Computing technologies and offers," the Proceedings of the 2011 First International Symposium on Network Cloud Computing and Applications, pp. 131-134 , 2011.

[12] Groth P., "A Distributed Algorithm for Determining the Provenance of Data, " the IEEE Fourth International Conference on eScience, pp. 166-173, 2008.

[13] IOzone, http://www.iozone.org, 2013.

[14] Virtio, http://www.linux-kvm.org/page/Virtio/ ,2013.

[15] Ceph filesytem , http://ceph.com/