International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:6, No:8, 2012

Reducing Power Consumption in Cloud Platforms using an Effective Mechanism

Shuen-Tai Wang, Chin-Hung Li, Ying-Chuan Chen

***Abstract*—**In recent years there has been renewal of interest in the relation between Green IT and Cloud Computing. The growing use of computers in cloud platform has caused marked energy consumption, putting negative pressure on electricity cost of cloud data center. This paper proposes an effective mechanism to reduce energy utilization in cloud computing environments. We present initial work on the integration of resource and power management that aims at reducing power consumption. Our mechanism relies on recalling virtualization services dynamically according to user's virtualization request and temporarily shutting down the physical machines after finish in order to conserve energy. Given the estimated energy consumption, this proposed effort has the potential to positively impact power consumption. The results from the experiment concluded that energy indeed can be saved by powering off the idling physical machines in cloud platforms.

***Keywords*—**Green IT, Cloud Computing, virtualization, power consumption.

I. Introduction

THERE is growing interest in reducing the power consumption of data centers and cloud computing environments. Cloud computing [1,2] is the access to computers and their functionality through different network modes like public, private and hybrid. The cloud users can request this access from a set of web service interfaces that manage and monitor a pool of computing resources including machines, network, storage, operating system (OS), application programs and development environments. When granted service requests, a part of the computing resources in the pool is dedicated to the requesting user until those resources are released. Moreover, users can not actually see or specify the physical location and organization of the equipment hosting the resources. Virtualization technology acts as a central component that can achieve the purpose of cloud platforms and services, and it is a promising approach to consolidating multiple services onto a smaller number of computing resources. A virtualized server environment allows computing resources to be shared among multiple performance-isolated platforms called virtual machines [3,4]. A virtual machine is a software implementation of a machine that executes related programs like a physical machine.

S.T. Wang is with the National Center for High-Performance Computing, Taiwan, R.O.C. (e-mail: stwang@nchc.org.tw).
C.H. Li is with the National Center for High-Performance Computing, Taiwan, R.O.C. (e-mail: oscar@nchc.org.tw).
Y.C. Chen is with the National Center for High-Performance Computing, Taiwan, R.O.C. (e-mail: ycc0301@nchc.org.tw).

Each virtual machine includes its own system kernel, OS, supporting libraries and applications. A hypervisor provides a uniform abstraction of the underlying physical machine, and multiple virtual machines can execute simultaneously on a single hypervisor. Decoupling of virtual machine from the underlying physical hardware is able to allow the same virtual machine to be started and run on different physical environments. Thus virtualization is seen as an enabler for cloud computing, allowing the cloud service provider the necessary flexibility to move and allocate the computing resources requested by the user wherever the physical resources are available.

Virtualization also enables on-demand resource provisioning model in which computing resources such as CPU, memory, and disk space are made available to applications only as needed and not allocated statically. So by dynamically provisioning virtual machines, consolidating the workload, and turning computers on and off as needed, the administrators can maintain the desired quality of service while achieving higher computer utilization and energy efficiency in virtualization cloud platform.

To do so, in this paper, we present an innovative way to reduce energy utilization in cloud platforms. In particular, our approach relies on recalling services dynamically onto appropriate amount of the computers according to user's request and temporarily shutting down the computers after finish in order to conserve energy. We present initial work on the effective integration of resource and power management system that aims at reducing power consumption such that they suffice for meeting the minimizing quality of service required by data centers or cloud computing environments. For smartly controlling power usage, new features will be added to cloud resource management system. Integrating extra remote power management ability for physical machine is also inevitable. Given the estimated energy consumption, the proposed effort has the potential to positively impact power consumption.

This paper explains how to customize the current cloud infrastructure and middleware to reduce total power consumption. The rest of this paper is organized as follows. Section 2 lists the background. Section 3 gives some details of the implementation. The result and analysis will be presented in section 4. Finally, section 5 presents the conclusion.

II. Background

A. Green Computing

Green Computing [5] is the study and practice of using IT resources more efficiency.

Today, Green Computing is becoming more and more urgent. The effort is to not only to reduce cost and increase access to technology, but also to lower the impact on our environments through recycling, new IC layout, system virtualization, etc. A number of focus areas and activities are mentioned to achieve this goal. Unlike IT vendors tending to invent lower power devices, here, we concentrate on two issues: energy-efficient computing and power management, and try to figure out what else we can do in the high performance computing cluster.

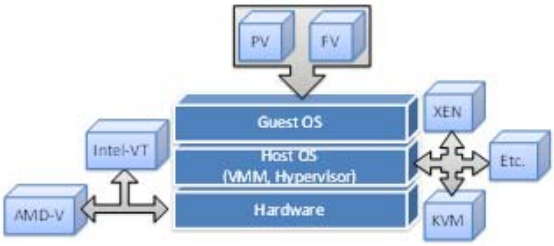

Fig. 1 Virtualization Architecture

B. Virtualization Architecture

Figure 1 shows the principal architecture of virtualization. Physical hardware resources were divided as virtual resources of virtualization platforms by the monitors, and those virtual resources were assigned to each virtual machine by different application requirements. Furthermore, The monitor provides each Guest OS a set of virtual platform interfaces that constitute virtual machines, acting as a bridge to connect between hardware and virtual devices. Instructions were delivered to hardware layer from virtual platform, which receives results from monitor of virtual machines. Each virtual platform will run independently, although physical resources were shared. By the way, the monitor has two kinds of model, hypervisor and virtual machine monitor (VMM) [6]. The main distinction between Hypervisor and VMM is that the former monitor runs above hardware layer directly with better performance than VMM, such as Xen [7] and VMware's ESX [8]. On the other hand, Microsoft's Virtual PC and VMware's Workstation adopt VMM as monitor of virtual platforms.

For Guest OS, it includes two main virtualization types [9]: para-virtualization (PV) and full-virtualization (FV), which can be both combined with hardware-assisted virtualization. The Guest OS is simulated by modified kernel of Linux with PV, but related devices are not emulated. Instead, all devices are accessed through light-weight virtual drivers offering better system performance and close to the physical machine. But the drawback is that guest kernels must be upgraded to provide new virtual system calls for the new services and all of guest OS must be compatible with the host OS

C. Virtual Machine

Virtualization technology is able to apply not only to subsystems but to a complete virtual system. To implement a virtual machine, software developers design a software layer to real machines to support the desired architecture. By providing one or more efficient virtual platforms, virtual machines have extended multi-processing systems of the past decade to become multi-environment systems as well. There are many kinds of virtual machine in the market, but not all virtual machines fit to build virtual platforms, so we choose the Kernel-based Virtual Machine (KVM) [10] to achieve our purpose. KVM is an open source software with GPL, developing by Qumranet company. KVM provide FV solution for Linux on x86 hardware containing virtualization extensions with Intel-VT or AMD-V, and the kernel component of KVM is encased in mainline Linux OS over version 2.6.20, hence the user can set up the virtualization environment of KVM when installing Linux OS conveniently.

Using KVM, we can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware devices, such as network card, disk, graphics adapter, etc. Besides, virtual machine was like a process in queuing system of Linux, and then manager can directly kill any virtual machines by control command.

D.IPMI

IPMI stands for Intelligent Platform Management Interface [11]. The IPMI specification defines a set of common interfaces to a computer system which system administrators can use to monitor system health and manage the system. It operates independently of the operating system and allows administrators to manage a system remotely even in the absence of an operating system or even if the monitored system is powered off but connected to a power source. With IPMI tool, administrators can know the detailed hardware health more easily. The most interesting feature is that IPMI operates independently of the Linux distribution. Administrators can manage compute nodes remotely even in the absence of management utility as long as ethernet and power cables connected. IPMI can also work when OS begins its operation and provide other advanced features when worked with IPMI management tool. We employ IPMItool toolkit to handle tedious tasks in our cloud platform.

E. STD and STR

Since frequently power-on and shutdown of physical machines would cause more electricity lost and long waiting time of booting. So we need a wise approach to make machine sleep and resume more quickly. Most modern Linux distribution (kernel 2.6.15 or later) use Advanced Configuration and Power Interface (ACPI) [12] for Linux power management. Suspend-to-Ram (STR) is ACPI S3 state where it puts your computer into a low-power state so that users can quickly resume Linux login session. All data will be stored into main memory during this state. One known drawback is that resume from Suspend-to-Ram will not always work as expected. It is still an ongoing function for Linux kernel developers and device vendors. Another sleep function in Linux is ACPI S4 state, which is known as Hibernate or Suspend-to-Disk (STD). From a power consumption point of view, when a computer is entering Hibernate, it acts the same as machine powered off.

It takes less electricity but could stay much longer in sleeping mode than Suspend-to-Ram. Since Hibernate stores all processes' context to hard disk, it will be more secure for applications and operating system. In some advance motherboards, the power source can even be removed without losing data. The only inconvenience is that Hibernate needs longer period of time to turn the computer back on.

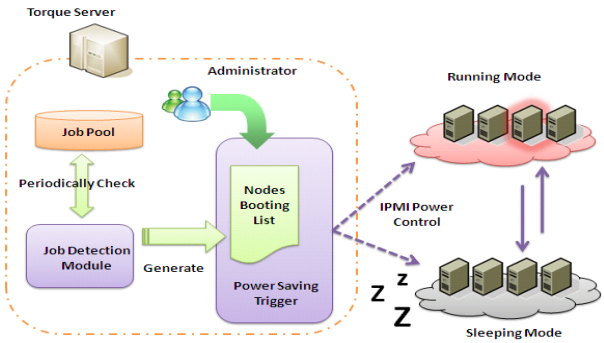

Fig. 2 Power Management Architecture

III. Implementation

Figure 2 is the architecture of our implementation. The core of this implementation is Power Management Wizard package, which wrapped the major components we developed are: Job Detection Module, Power Saving Trigger, and Remote Power Control. These components will be made a description in detail as followings.

A. Resource Manager

For resource management of cloud platform, we developed a resource manager providing control over virtualization request jobs to guarantee the fairness of using the physical machines, priority escalading, and resource partitioning. We also developed a special module called Job Detection Module to detect the actual virtualization job loading. When users start their virtualization jobs into resource manager system, all jobs will be queued as 'W' state at first. This customized process will calculate the how many physical processors that jobs need and written it to Nodes Booting List. This list also acts as load history provider for Power Saving Trigger. According to load history information, trigger can smartly decide which kind of Sleeping Mode the compute nodes should enter. If recent load were lower than average, cold shout-down command will be used for compute nodes. On the other hand, if platform was busy recently, hibernate or suspend command will be sent to them.

B. IPMI Integration

Modern motherboards come with BMC [13] card or built-in chips support IPMI protocol. This protocol makes cluster remote management possible. Like most manufacturers tend to add proprietary extensions to their implementations of IPMI. In this project, we also integrate the IPMItool toolkit in Power Saving Trigger to support batch loading node list file and power on/off a group of predefined compute nodes.

The format of node list file is simply ipmi-ip and ethernet-ip pair. The automatic mapping of these two IP address also has been implemented and helps Power Saving Trigger to identify the target machine. Use the following example feeding the node profile data into Power Saving Trigger. These nodes will be viewed as target of commands.

```
$ ipmitool -I open batch addnode < ipmi-node.dat
```

For practical reasons, these additional parameters will be helpful when administrators need to do batch operation (to query, shutdown, etc.) in Power Saving Trigger. The following are a few examples of batch operation.

```
$ ipmitool –I open batch flushnode
$ ipmitool –I open batch listnode
$ ipmitool –I lan batch power on
$ ipmitool –I lan batch power off
$ ipmitool –I lan batch power status
```

C. Remote Power Control

In order to avoid the extra power consumption during booting stage, sometimes entering sleep mode instead of cutting off real power supply is necessary. Hibernate or Suspend could offer great power savings since no power is consumed in this state. Besides wrapping IPMI protocol in this development, we also have to add DSH (Distributive SHell) [14] to Suspend-to-Disk/Ram for better remote control. There are three power-off methods we developed in Power Saving Trigger for switching from "Running Mode" to "Sleeping Mode". They are e-shutdown, e-suspend and e-hibernate commands for Linux console. Table I shows the comparison of these commands. These Power-Off methods can satisfy common Linux based clusters in despite of hardware variations.

TABLE I
Power-Off Strategies in Power Saving Trigger

Command Name	e-shutdown	e-suspend	e-hibernate
Wrapped Mechanism	Power Off via IPMI	Suspend via DSH	Hibernate via DSH
Response time	Quick	Medium	Slow
Need motherboard support	Yes	Yes	Yes
Need power source connected	Yes	Yes	No
Need additional IP address	Yes	No	No
Data & System Integrity	Low	Medium	High

The Power Saving Trigger will refer to history load information in Node Booting List and decides which action it should take next time. When load of recent platform usage is lower than 60 percent, the Power Saving Trigger will adopt IPMI protocol to power off compute nodes.

In contrast, if the loading is higher 60 percent (it means platform was busy recently), it will choose e-suspend or e-hibernate to force compute nodes to sleep. Due to original protocol constrains, if the Power Saving Trigger uses IPMI way to power off compute nodes, then it would use either IPMI or WOL [15] to power-on the compute nodes. Unlike cold shutdown, if the Power Saving Trigger use e-suspend or e-hibernate command to enter "Sleeping Mode", forking remote command via SSH protocol is the only way to power on machines next round. For flexible management, Cluster Power Management Wizard supplies two new commands to do power-on via network for compute nodes. They are e-poweron and e-wakeonlan listed in Table II.

TABLE II
POWER-ON STRATEGIES IN POWER MANAGEMENT WIZARD

Command Name	e-poweron	e-wakeonlan
Wrapped Mechanism	Power On via IPMI	Wake On Lan
Need motherboard support	Yes	Yes
Need power source connected	Yes	No
Need additional IP address	Yes	Yes
Need Collecting MAC address	No	Yes

IV. RESULTS AND ANALYSIS

This section describes our experimental setup, including the testbed, energy evaluation on real facilities, and power consumption analysis.

A. The Testbed

Formosa 3 Cloud Cluster [16] is a 64bits high-performance Beowulf cluster located within Southern Business Unit of the National Center for High Performance Computing (NCHC) [17]. It consists of 76 IBM X3550M3 servers as its compute nodes. This self-made cluster was designed and constructed by the 'HPC Cluster Group' at NCHC for cloud IaaS (Infrastructure as a Service) service and came online in 2011. Each node has two Dual-Core Intel Xeon 5660 2.8GHz processors and 48GB of DDR3 registered ECC SDRAM. All nodes were connected on the InfiniBand high speed network and a private subnet with 1000 Mbits/s Gigabit Ethernet. An additional 4 nodes are used as front ends to interface with cluster, and 4 nodes as storage for the user file systems by Parallel File System. To determine the total power used by Formosa 3, we could refer to the manufacturer's specification. Theoretically, total power consumption produced by this cluster is as Table III:

TABLE III
DEVICE WATTAGE RATINGS FOR FORMOSA 3 CONFIGUATION

	Total Device	Watts per Device	Total Watts
IBM X3550M3	80	411	32880
ARA Stor A316i	4	1050	4200
Nortel BayStack 5510 Gigabit Switch	6	135	810
Voltaire ISR 9288 InfiniBand Switch	1	2340	2340

The maximum wattage for this cluster's active devices is estimated at 40,230W = 40.23KW. From its inception, the Formosa 3 has embarked on open source movement.

B. Energy Evaluation of Formosa 3

The power for Formosa 3 server racks are offered by Power Distribution Unit in NCHC machine room. From the electricity monitor on each rack, we can easily measure the Ampere value per hour. Table IV shows the rack arrangement and daily power consumption measured of Formosa 3. Average Ampere multiplied by 220 (voltage) will get the average watt per hour of each rack. Notice that four racks may be divided into three groups. Both Rack 1 and Rack 2 contained pure compute nodes, so energy intake of these two racks was higher than the rest. The converse holds for Rack 3, this rack consumes least electricity in rating. The Rack 4, holding both compute and storage nodes, consumes medium electricity. To understand the impact, we deployed our mechanism on Formosa 3 cloud platform that runs actual virtual machines and investigate different workload scenarios. Experiment input data were two job workloads during August and October.

TABLE IV
AVERAGE WATTS PER DAY

	Staged Hardware	Quantity	Average Ampere	Average Watts
Rack 1	Physical Machine	30	46.72	10278.4
Rack 2	Physical Machine	30	44.98	9895.6
Rack 3	IB Switch	1	11.21	2466.2
	Gigabit Switch	6		
	Login Node	4		
Rack 4	Storage Node	4	28.87	6351.4
	Physical Machine	12		

C. Power Consumption Analysis

In this section, there are four cases arranged for power consumption evaluation. They are: (1). Power consumption of physical machines always be powered on, (2). Power consumption of simply using e-shutdown command test, (3). Power consumption of simply using e-hibernate command test, (4). Power consumption of combining e-shutdown and e-hibernate test. Case1 represents s regular cluster in the world that all its nodes are always keeping ready. Case 2 and Case 3 and Case 4 are special scenes while adding power management function to the cloud platform. From the log data from August 2011, 2713 virtualization job requests completed during the experiment time. For the October 2011, 3934 virtualization jobs workload, a different pattern, frequently submission behavior occurred. By running two job workloads through four scenarios, we show that Power Management Wizard can have a large positive impact on power conservation. Table V summarizes power consumption of four cases. Bottom row in Table VI designates how much energy could be saved when compared to Case 1.

TABLE V
AVERAGE ENERGY USAGE (IN WATTS) OF LOW FREQUENCY USE

Aug. 2011	no shutdown	e-shutdown	e-hibernate	e-shutdown & e-hibernate
Rack 1	9412.3	6342.3	6943.1	6312.8
Rack 2	9328.4	6412.1	7313.6	6439.7
Rack 3	1994.7	1853.6	1989.8	1873.2
Rack 4	5989.1	3811.0	4112.3	3989.1
Total	26733.5	18419	20358.8	18614.8
	Case 1	Case 2	Case 3	Case 4
Saved	0	8314.5	6374.7	8118.7

TABLE VI
AVERAGE ENERGY USAGE (IN WATTS) OF HIGH FREQUENCY USE

Aug. 2011	no shutdown	e-shutdown	e-hibernate	e-shutdown & e-hibernate
Rack 1	9871.4	8361.6	7113.8	8113.2
Rack 2	9634.1	8863.6	7314.1	8601.3
Rack 3	2111.3	1964.3	1861.0	2004.6
Rack 4	6322.7	4632.2	4432.2	4318.7
Total	27939.5	23821.7	20721.1	23037.8
	Case 1	Case 2	Case 3	Case 4
Saved	0	4117.8	7218.4	4901.7

V. CONCLUSION

With the Green IT concept gradually accepted by consumer and industry, those cloud computing environments or so-called data centers, used to emphasize on mass production and time efficiency, also realize this worldwide trend.

The cloud computing machines generally take the majority power consumption in a machine room. Rather than buying off-the-shelf, power-efficiency machines from vendors, this paper gives a power saving mechanism for how to greenly using current cloud computing facilities. The approaches in this paper were to describe a harmless solution and a process that can be used on these devices. We explored the effect and affect of present initial results. By powering off idle physical machines, it can significantly save more energy than always keeping these physical machines running. Furthermore, no matter how the load of cloud platform changes over time, this power management wizard could adapt different power management strategies for physical machines. All techniques or ideas mentioned in this paper could be applied to other infrequently used servers to prevent energy waste in cloud data centers.

REFERENCES

[1] I. Foster, Y. Zhao, I. Raicu, and S. Lu. “Cloud Computing and Grid Computing 360-Degree Compared,” IEEE Grid Computing Environments Workshop, pp. 1-10, 2008.
[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, and M. Zaharia, “Above the Clouds: A Berkeley View of Cloud Computing,” 2009.
[3] R. A. Meyer and L. H. Seawright, “A Virtual Machine Time-Sharing System,” IBM Systems Journal, vol. 9, no. 3, 1970.
[4] R. P. Goldberg, “Architecture of Virtual Machines,” National Computer Conference Proceedings, AFIPS Press, vol. 42, pp. 309-318, June 1973.
[5] C. MacKinnon, “Green Computing In The Data Center,” Processor Editorial Article, Vol. 29, 2007.
[6] G. J. Popek, and R. P. Goldberg. “Formal Requirements for Virtualizable Third Generation Architectures,” Communications of the ACM, vol. 17, pp. 412-421, 1974.
[7] Xen hypervisor, Available at: http://www.xen.org/
[8] VMware virtualization, Available at: http://www.vmware.com/
[9] W. Chen, H. Lu, L. Shen, Z. Wang, N. Xiao, and D. Chen. “A Novel Hardware Assisted Full Virtualization Technique,” The 9th International Conference for Young Computer Scientists, pp. 1292-1297, 2008.
[10] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. “kvm: the Linux Virtual Machine Monitor,” In Proceedings of the Linux Symposium, vol. 1, pp. 225-230, 2007.
[11] IPMI, Intelligent Platform Management Interface, Available at: http://www.intel.com/design/servers/ipmi/
[12] ACPI, Advanced Configuration and Power Interface, Available at: http://www.acpi.info
[13] H. Zhuo; J. Yin, A. V. Rao, "Remote Management with the Baseboard Management Controller," Available at: http://www.dell.com/downloads/global/power/ps4q04-20040110-Zhuo.pdf
[14] DSH, The Distributed Shell, Available at: http://dsh.sourceforge.net
[15] Wikipedia, WOL, Wake-on-Lan, Available at: http://en.wikipedia.org/wiki/Wake-on-LAN
[16] NCHC Formosa 3 Cloud Cluster, Available at: http://formosa3.nchc.org.tw/
[17] NCHC, National Center for High-performance Computing, Available at: http://www.nchc.org.tw