A Power-Aware Cloud Architecture with Smart Metering

Che-Yuan Tu, Wen-Chieh Kuo,
Wei-Hua Teng, Yao-Tsung Wang
National Center for High-Performance Computing,
No. 7, R&D Rd. VI Hsinchu Science Park,
Hsinchu, Taiwan
{rider, rock, wade, jazz}@nchc.org.tw

Steven Shiau
Engineering Science Dept.,
National Cheng Kung University,
No.1, University Road, Tainan City 701, Taiwan
steven@nchc.org.tw

***Abstract*—With the growing interest of cloud computing and carbon emission reduction, how to build energy efficient cloud architecture becomes a crisis issue for service providers. In this paper, we propose a power-aware cloud architecture based on DRBL (Diskless Remote Boot in Linux), cpufreqd and xenpm. We also introduce a low-cost smart metering system based on open hardware Arduino board. Composing with existing technique, such as Dynamic Voltage Frequency Scaling (DVFS), ACPI, diskless design and RAM disk storage, our experiment results show that this architecture will reduce energy consumption from 4 to 11% when running CPU-intensive applications. In conclusion, this paper reveals that service providers could benefit from diskless design and RAM disk storage if their applications are CPU-intensive.**

Keywords-Arduino; Cloud Computing; Diskless Remote Boot in Linux (DRBL); Green Computing; Xen

I. INTRODUCTION

As the cloud computing becomes the most pervasive business slogan, there are more and more various services that could be accessible from any internet-enabled devices over the internet. To achieve the Near-Zero-Downtime internet service and its performance, the cloud platforms composed from a pool of computer resources are typically deployed into clusters of massive number of servers hosted in data centers. However, large amount of stable energy and power consumption play a vital role in providing the cloud platform needs for IT industries. According to the EPA report, servers and data centers consumed about 61 billion kilowatt-hours (kW·h), and it was 1.5% of the total U.S. electricity consumption for about $4.5 billion electricity cost in 2006. Furthermore, trends show that the power consumption still keep growing at 18% annually, and it also estimates that the power consumption of data centers could nearly double by 2011 [7]. Besides, most of the climatologists believe that there should be a strong relationship between greenhouse effect and global warning, and the ringleader maybe the large amount of carbon dioxide emission. The EIA report also shows that 98% of carbon dioxide emission may be directly attributed to energy consumption [6]. Globally we are facing such crisis and have to make an effort to improve energy efficiency and avoid waste.

In order to provide an easy way to build an energy efficient cloud platform, we had already built the ClassCloud project [3]. We proved that it is suitable for CPU-intensive applications and provides an energy efficient architecture based on diskless design from the experiment result of the E2CC [3]. According to the technical white paper from Uptime Institute, it obviously shows that server gains 50% chance of system failure for each 10 ^{o}C increase over 20 ^{o}C [12], and there are also similar statistics for hard disk lifetimes [1][4]. That's one of the reasons why the diskless design of DRBL benefits this architecture, and users can focus on server maintenance and recovery without considering client nodes. Besides, all of the data related to client nodes like boot image files, configuration files are all centralized on DRBL server because of the diskless design. Therefore, the diskless design can not only provide an energy efficient architecture but also ease of management.

Going a step further, **power management** and **smart metering system** are two main features of this architecture, and this paper presents how to enhance it with power saving and smart meter by using Dynamic Voltage Frequency Scaling (DVFS) and Arduino based on the ClassCloud project. The discussion of how the DVFS can save energy and how to build a power meter for smart metering is to be concerned in following chapters. Basically, way for achiving power management and energy efficiency is to find out where and why waste happens and determining how to avoid it [10]. Traditionally, systems have been designed to achieve maximum performance for the workload, but report from NRDC indicates that servers still use 69-97% of total energy when idle [8]. Therefore, we can avoid waste and reduce power consumption by scaling down CPU frequencies and changing throttling levels via DVFS to minimize the processor power dissipation. DVFS is a technique in computer architecture whereby the frequency of a microprocessor can be automatically adjusted "on-the-fly" to conserve power and to reduce the amount of heat generated by the chip. With the hardware support of Intel® SpeedStep and AMD PowerNow!™ technologies, modern processors can be set in all available frequencies and throttling levels on different voltage. Hence, we adopt it into this cloud architecture to reduce processor power dissipation, and set demand frequencies and throttling levels at idle time or application runtime on

different phases. One of the open source tool named cpufreqd [5] is used to help users to set demand frequencies for dedicated applications automatically and lower down after finishing jobs according to the rules defined by users.

On the other hand, smart metering is the next generation electricity measurement technique to optimize power efficiency, and users can monitor daily power usage and choose the most suitable pricing model to evaluate their electricity cost within different period of time. In the cloud architecture, we take Arduino as the power meter device and integrate with current transducer and cloud server to become a smart metering system which provides power consumption calculation, electricity cost evaluation and power outage notification. However, the reason for choosing Arduino is that it's an open source hardware which allows users to design and build their devices elastically with lower cost.

In brief, this cloud architecture provides a power-aware architecture to reduce processor power dissipation and helps to avoid waste from machine idle time. Besides, the energy consumption experiment proves that running CPU-intensive applications in diskless mode with RAM disk storage can save more power than in diskfull within all different CPU frequnecies and throttling levels. We can say that the diskless design and RAM disk storage can really take effect on energy saving, but the performance is not what we expected. Besides, in order to provide an easy way for users to monitor system status and electricity information from machines, so the smart metering system provides a web-based interface via Ganglia.

II. BACKGROUND

A. Arduino

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software [2]. Based on open source hardware, it opens all hardware design and related information to everyone, so users can freely design and implement their ideas. Arduino is composed of a single-board microcontroller and multiple I/O support, and it can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. Furthermore, users can transfer data to any network supported devices through communication mechanisms such as RJ-45, IEEE 802.11 and IEEE 802.15.4-compliant wireless networks.

B. DRBL

Diskless Remote Boot in Linux (DRBL) is an open source solution to managing the deployment of the GNU/Linux operating system across many clients [9]. DRBL supports lots of popular GNU/Linux distributions, and it is developed based on diskless and systemless environment for client machines. DRBL uses PXE/Etherboot, DHCP, TFTP, NFS and NIS to provide services to client machines so that it is not necessary to install GNU/Linux on client hard drives individually [9]. Users just prepare a machine as the DRBL server, and follow the DRBL installation wizard to configure and dispose the environment for client machines step by step. That's really an easy job to deploy a DRBL environment on clustering systems even for a GNU/Linux beginner, hence cross-platform and user-friendly are the key factors that make the DRBL become a superior clustering tool.

C. Xen

Xen is an open source virtual machine monitor (VMM/Hypervisor) for x86, x86_64, IA64, PowerPC, and other CPU architectures [14]. According to Paul Barham and Boris Dragovic et al. [11], Xen uses para-virtualization to avoid performance losses of full virtualization, and brings near-native performance to VMs (virtual machines). Besides, Xen also supports full virtualization through support from the hardware (Intel-VT, AMD-V) and live migration for Near-Zero-Downtime applications and services. On the other hand, it brings new features to change CPU frequency and voltage through the acpi-CPUfreq driver in Xen version 3.4.x [15], so users can change their VMs into different CPU C/P states according to users' preference via xenpm tools.

III. THE CLOUD ARCHITECTURE

In most of the times, PC classrooms in education organizations and academic institutes are lying idle, so we want to bring the cloud computing concept into them for improving utilization. In our research center, we use DRBL/Clonezilla as management tools to turn our PC classrooms into different demands for various courses, and now we want to make use of virtualization technology to build an experiment environment for cloud computing. Table 1 shows the hardware specifications and software list of this architecture.

TABLE I. HARDWARE SPECIFICATIONS AND SOFTWARE LIST

Hardware	Software
Intel® Core™ 2 Quad CPU Q6600 @ 2.40GHz	Debian GNU/Linux 5.0.4 (lenny)
2.5GB RAM	Kernel 2.6.26
160GB Hard Drive	DRBL 1.9.5-79
Intel 82571EB Gigabit NIC	Xen 3.4.0
Hardware (Network switch)	
Linksys SLM2048 48-port 10/100/1000 Gigabit Switch	

A. Test Environment

DRBL is a superior central management tool for clustering system, and it provides an easy way to deploy a unified environment for a variety of purposes. In order to make PC classrooms as a cloud computing testbed, it should provide capabilities to deploy and manage easily. However, we want to improve the original architecture with better power management and smart metering system, so

there are some techniques, like Arduino, DVFS and Ganglia that we can use for enhancing the architecture. First, the DVFS technique helps to change available CPU frequencies and voltage on different purposes according to users' demands or other priorities, and cpufreqd is the tool that helps users to set demand frequencies for dedicated applications automatically and lower down after jobs finished. Second, we develop an economical and practical power metering device by using Arduino and build a power monitoring system with Ganglia through RS-232 or ZigBee. Thus, these two components which are power monitoring and metering system are rephrase of this architecture.

Fig. 1 shows the test environment. The cloud server is responsible for the deployment and management in clustering system. Because of the diskless design, clients get all of the necessary files including booting files, configuration files and root file systems from cloud server through NIS, NFS and TFTP protocols. Actually, the central management mechanism of this architecture helps users to deploy and manage clustering system more easily, and it also saves lots of time on system maintenance and recovery. Furthermore, we modify the vanilla kernel for supporting Xen and use DRBL to deploy elastically all kinds of VMs for different purposes. Besides, we collect all of the related data such as CPU temperature, CPU voltage and electricity usage into Ganglia from those machines which are available on the architecture by using lm-sensors and power meter. Besides, the cloud server provides a web-based monitoring user interface powered by Ganglia, so users can monitor their clustering system status on this architecture.

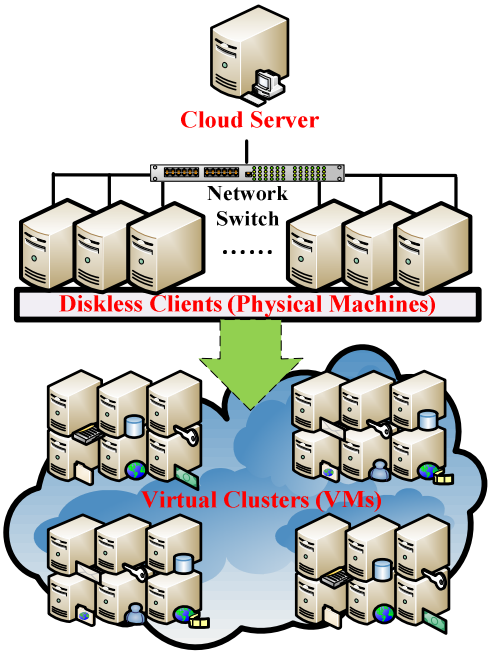


Figure 1. Cloud Test Environment.

B. *Software Stacks*

Fig. 2 shows the software stacks of this architecture. The Xen hypervisor, Xen patched kernel and kernel modules belong to kernel space. Xen hypervisor is the bottom layer, and it handles hardware access between host and guest/VMs. On the other hand, the DRBL, Ganglia, PMtools and Xen tools are the following four components in user-space: (1) DRBL can automatically bind these Linux daemons (TFTPd, NISd, DHCPd, and NFSd) to build up a cluster environment and offer management utilities to clusters ; (2) Ganglia monitor provides two daemons for both server and client which are gmetad and gmond, and it helps to monitor system status on clustering system ; (3) PMtools is the main component for power management, among them xenpm is the control tool based on DVFS technique for Xen virtual machines, and cpufreqd is the similar one for physical machine ; and (4) Xen tools is a collection of simple scripts which allow users to easily create new guest Xen domains upon GNU/Linux host.

IV. SMART METERING SYSTEM

Smart metering is the next generation electricity measurement technique to optimize power efficiency. According to Tarry Singh and Pavan Kumar Vara, a smart metering device is usually composed of real-time sensors with power outage notification and power quality monitoring capabilities [13]. Actually, a good design of smart metering should provide lots of real-time pricing models for distinct regions according to different electricity rates in different periods of time.

A. *Smart Metering Device based on Arduino*

There are lots of power measurement tools, like PDU (Power Distribution Unit), digital multimeter (DMM) and current meter which are not price accessible to anyone. Besides, most of these devices are designed with restricted capabilities and scalabilities in a closed platform. In order to build an elastic and low-cost power meter, we consider using the Arduino open-source electronics prototyping platform. Arduino is composed of a single-board microcontroller and multiple I/O support, and it can sense the environment by receiving input from a variety of sensors. That's the reason why we choose Arduino as the power meter.

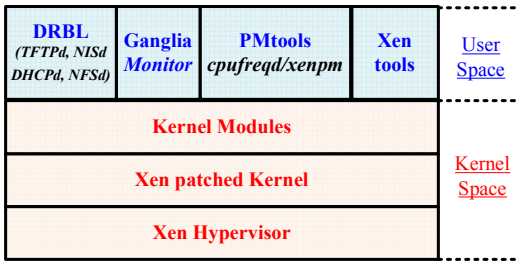


Figure 2. Software Stacks.

The key characteristics of Arduino:

- **Elasticity:** Users can fetch monitoring data through multiple transmission protocol, like Ethernet, RS-232 and ZigBee by choosing to use different sensors or modules.
- **Extensibility:** There are lots of electrical components and modules that support Arduino. Users can design different kinds of devices that meet their requirement with multiple I/O support.
- **Cost:** The Arduino development board and related components are cheap, so users can develop such a device with lower cost than buying a finished product separately.

B. *Smart Metering Architecture*

The smart metering architecture provides the following functionalities which are power consumption monitoring, electricity pricing evaluation and power outage notification. Owing to measure the current from the circuit breaker or the power line, there should be a current transducer that connects between the circuit breaker or the power line and the Arduino board. The current transducer transfers the real-time measured alternating current (AC) to Arduino board, and transforms measured data from analog to digital. Thus, there is a transformation module which is responsible for real-time current transformation and delivers transformed data to the cloud server via an RS-232 connection. Besides, there is a pricing module which helps to calculate electricity cost of this architecture, but users have to set the local electricity rates first according to different periods of time to calculate the actual cost. Our cloud server uses an uninterruptible power supply (UPS) to avoid sudden outage. Once the cloud server receives a current value of nearly zero from the smart meter for a continuous time, the cloud server will send a notification e-mail to system administrators. Besides, users can also check the power and system status through a web-based monitoring interface via Ganglia. Fig. 3 is the architecture of the smart metering system.

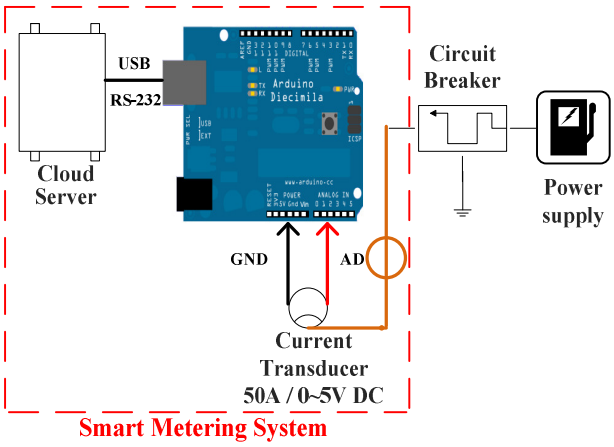


Figure 3. Smart Metering Architecture.

C. *Smart Metering Transformation Model*

There are three parts of unit conversion to lead out the final formula. The first step is to transform the measured AC current into DC voltage via the current transducer, so it converts measured AC current at the range from 0 to 50(A) into DC voltage ranged from 0 to 5(V) proportionally. The transformation in the first step can be expressed as the following equation:

$$V_{DCOut} = \frac{I_{ACIn}}{10} \tag{1}$$

V_{DCOut}: *DC voltage output from Current Transducer*
I_{ACIn}: *AC current input from power line*

The Arduino provides six analog inputs, and resolution of the Arduino analog-to-digital converter (ADC) is 10-bit. Thus, the ADC can encode an analog input to one in 1024 different levels according to measured voltage, since 2^{10} = 1024. The second transformation step is to convert the value of V_{DCOut} at the range from 0 to 5(V) from analog into digital within the ranges from 0 to 1023. Therefore, the second step can be expressed as the following equation:

$$VAD = \frac{1024}{5} V_{DCOut} \tag{2}$$

VAD: The Transformed value from Arduino ADC

The third step is to convert the value of VAD back into the I_{ACIn}. Because the AC current is at the range from 0 to 50(A), so this equation helps to convert the transformed value from Arduino ADC back into the AC current. The transformation of the third step can be expressed as following equation:

$$I_{ACIn} = \frac{50}{1024} VAD \tag{3}$$

Finally, the formula of this transformation model can be calculated by using this equation:

$$Current(A) = Value_{Arduino} \times \frac{5}{1024} \times \frac{50}{5} \tag{4}$$

$Value_{Arduino}$: *Output value from Arduino*

V. Energy Consumption Experiment

In order to prove the diskless design and power management that really take effect on this architecture through DVFS, we consider designing a CPU-intensive experiment to calculate the total energy usage in different conditions (CPU frequencies & throttling levels) under diskfull and diskless environments. The reason for choosing CPU-intensive applications for this experiment is because of the result of E2CC [3].

A. *Experiment Design*

The main purpose of this energy consumption experiment is to prove that the diskless design can save more energy than diskfull, and better performance is to be expected. But before that, it is necessary to check out that whether the Intel® SpeedStep or AMD PowerNow! ™ is supported or not on your machines. According to the Table 1 in Chapter 3, our machines are equipped with Intel® Core™ 2 Quad Q6600 processors with maximum frequency at 2.40GHz, and it can be scaled between 2.40GHz and 1.60GHz. Besides, the processor also provides eight CPU throttling levels to be scaled based on ACPI "T" states, but it is generally useless for power consumption reduction nowadays. In recent years, most of the power reduction solutions use frequency scaling and ACPI "C" states to deal with such problems, but we want to discuss the energy consumption within different frequencies and throttling levels under diskfull and diskless environment in this paper.

Fig. 4 shows the experiment architecture. This architecture includes fifteen diskless clients, one network switch and a smart metering system. In order to choose a CPU-intensive application for this experiment, we pick one of our most common cases which use FFmpeg to convert AVI videos to H.264/MPEG-4 format, and two computer animated films named Big Buck Bunny and Elephants Dream from Blender Foundation are selected for this experiment. The reason for choosing Big Buck Bunny and Elephant Dream for our experiment is that these two films belong to the open source movie project without any copyright contention. The experiment is to calculate energy consumption and performance efficiency when converting these two AVI videos to H.264/MPEG-4 format in three different CPU frequencies (2.40GHz, 1.60GHz and Dynamic) and eight throttling levels (T_0～T_7) under diskfull and diskless environments. We choose to run on 2.40GHz or 1.60GHz manually and use dynamic frequency controlled by cpufreqd.

B. *Experiment Analysis and Discussion*

There are two cases which are referred to Case 1: Local boot with hard disk (Diskfull), and Case 2: PXE boot from cloud server without hard disk (Diskless) and RAM disk is to be used for storage. Fig. 5 and Fig. 6 present the energy consumption in kilowatt-hour (kW·h) and saving percentage both in Case 1 and Case 2 within three different frequencies and eight throttling levels. It obviously shows that Case 2 saves more energy than Case 1 ranged from 4 to 11% unequally all the time. Therefore, the diskless design and RAM disk storage of the cloud architecture really take effect on energy saving. Furthermore, Fig. 5 and Fig. 6 also show the performance difference in time saving percentage both in Case 1 and Case 2 within three different frequencies and eight throttling levels, we can see that Case 2 has better performance than Case 1, but the performance is not what we expected in both cases. In fact, even though running on lower frequency may save more power than higher frequency, but the total energy consumption may waste more in most of the times. That's because running on lower frequency takes more time to finish jobs than higher frequency. Therefore, the best way to improve energy efficiency is to run applications at higher frequency and reduce frequency at idle time when job finished. Actually, it's not always the best choice to run any applications at the highest frequency, so it needs to take times to find and optimize the best frequency to meet both energy efficiency and better performance. Of course, users can also choose to use dynamic frequency which is controlled by cpufreqd automatically, and it can also help those who care about energy saving without having time to experiment.

VI. CONCLUSION

In recent years, concept of energy saving and carbon reduction are popular and rising issues of network economics. In this paper, we build a power-aware cloud architecture to help those who want to easily build a cloud computing platform based on our architecture, and it provides energy efficient and power aware capabilities by using diskless design, RAM disk storage and cpufreqd/xenpm tools. The cpufreqd helps to use higher CPU frequency for dealing with dedicated applications automatically according to the usage rules from cpufreqd configuration on physical machines, and xenpm helps to adjust virtual CPU frequencies on virtual machines. These two tools help to finish jobs efficiently by using higher CPU frequency and avoid power dissipation at idle time when jobs finished. Both of these two tools are known as DVFS technique based on ACPI.

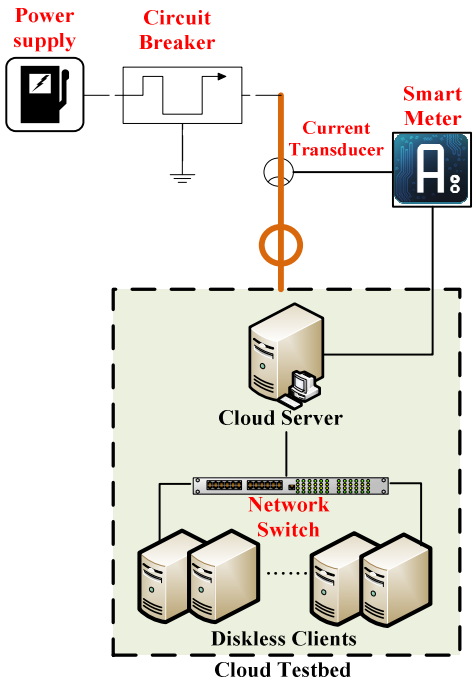


Figure 4. Experiment Architecture.

The main purpose of this energy consumption experiment is to prove that the diskless design can save more energy than diskfull, and better performance is to be expected by using RAM disk. According to the experiment results of energy consumption, it shows that the diskless design and RAM disk storage really take effect on energy saving at the range from 4 to 11% when running CPU-intensive applications, but the performance is not as we expect. We believe that the main factors for energy saving are the diskless design and RAM disk, although the performance difference between diskfull and diskless is not noticeable. Therefore, users can expect that using diskless design and RAM disk help to save more energy than diskfull. But if they expect better performance, it should take times to optimize applications case by case with different ways such as application algorithm, application/system scheduling etc.

Besides, we make use of the Arduino board to build an elastic and low-cost smart metering system which is composed of Arduino board, current transducer and cloud server for the purposes of power management and monitoring. The smart metering system helps to monitor power usage and electricity cost in soft real-time via Ganglia and notifies power outage by sending an e-mail to system administrators. In the future, we want to improve the smart metering system with location-based services (LBS) that helps to calculate electricity cost according to different regions and electricity rates, so it helps to adjust measures to local conditions.

REFERENCES

[1] Anderson, D., Dykes, J., and Riedel, E. More than an interface: SCSI vs. ATA. In Proceedings of the Second Usenix Conference on File and Storage Technologies (San Francisco, CA, Mar. 31--Apr. 2, 2003), 245--256.

[2] Arduino Website. [Online]. Available: http://www.arduino.cc/

[3] Che-Yuan Tu; Wen-Chieh Kuo; Yao-Tsung Wang; Steven Shiau, "E2CC: Building energy efficient ClassCloud using DRBL," in Grid *'09: Proceedings of the 10th annual international conference on Grid Computing*. Banff, AB, Canada: IEEE Computer Society, 2009, pp. 189-195.

[4] Cole, G. Estimating Drive Reliability in Desktop Computers and Consumer Electronics. Tech. Paper TP-338.1. Seagate Technology, Nov. 2000.

[5] Cpufreqd website. [Online]. Available: http://www.linux.it/~malattia/wiki/index.php/Cpufreqd

[6] Emissions of Greenhouse Gases in the United States. Energy Information Administration. Report, 2006. [Online]. ftp://ftp.eia.doe.gov/pub/oiaf/1605/cdrom/pdf/ggrpt/057306.pdf

[7] EPA Report to Congress on Server and Data Center Energy Efficiency. [Online]. http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Report_Exec_Summary_Final.pdf

[8] Natural Resources Defense Council "Recommendations for Tier I ENERGY STAR Computer Specification". [Online]. http://www.energystar.gov/ia/partners/prod_development/revisions/downloads/computer/RecommendationsTierICompSpecs.pdf

[9] NCHC. *Diskless Remote Boot in Linux (DRBL)*. [Online]. Available: http://drbl.sourceforge.net/

[10] Parthasarathy Ranganathan, "Recipe for efficiency: principles of power-aware computing," in Communications of the ACM, 2010, Volume 53,Issue 4,pp. 60-67.

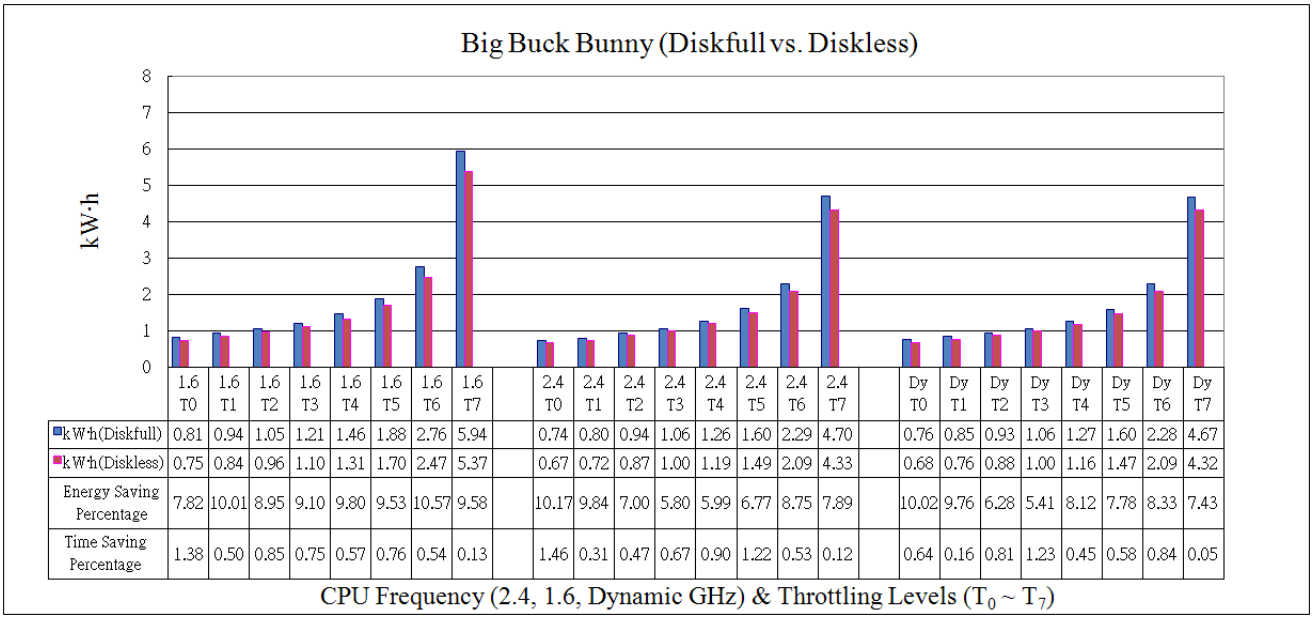
[11] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," *in SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles.* New York, NY, USA: ACM Press, 2003.

[12] Sullivan, R.F. Alternating Cold and Hot Aisles Provides More Reliable Cooling for Server Farms. White paper, Uptime Institute, 2000; http://www.dataclean.com/pdf/AlternColdnew.pdf

[13] Tarry Singh; Pavan Kumar Vara, "Smart Metering the Clouds," in Grid *'09: Proceedings of the 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*. Groningen, Netherlands: IEEE Computer Society, 2009, pp. 66-71.

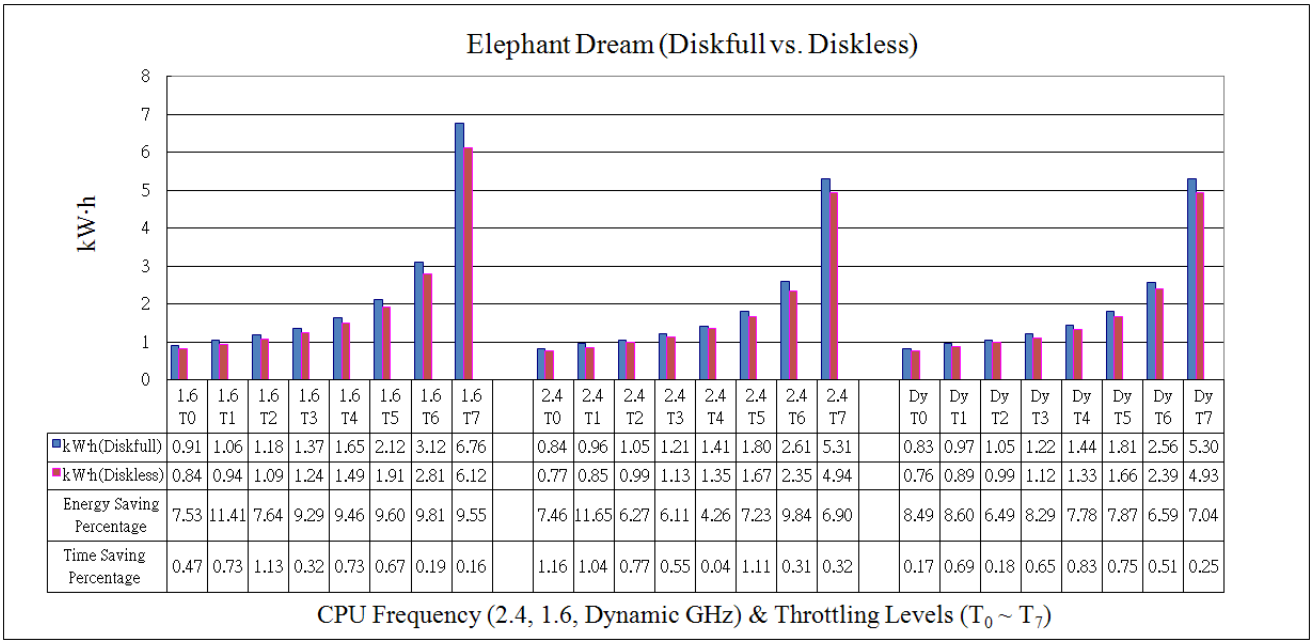
[14] Xen website. [Online]. Available: http://www.xen.org/

[15] Xen Wiki. [Online]. Available: http://wiki.xensource.com/xenwiki/xenpm



	1.6 T0	1.6 T1	1.6 T2	1.6 T3	1.6 T4	1.6 T5	1.6 T6	1.6 T7		2.4 T0	2.4 T1	2.4 T2	2.4 T3	2.4 T4	2.4 T5	2.4 T6	2.4 T7		Dy T0	Dy T1	Dy T2	Dy T3	Dy T4	Dy T5	Dy T6	Dy T7
kW·h(Diskfull)	0.81	0.94	1.05	1.21	1.46	1.88	2.76	5.94		0.74	0.80	0.94	1.06	1.26	1.60	2.29	4.70		0.76	0.85	0.93	1.06	1.27	1.60	2.28	4.67
kW·h(Diskless)	0.75	0.84	0.96	1.10	1.31	1.70	2.47	5.37		0.67	0.72	0.87	1.00	1.19	1.49	2.09	4.33		0.68	0.76	0.88	1.00	1.16	1.47	2.09	4.32
Energy Saving Percentage	7.82	10.01	8.95	9.10	9.80	9.53	10.57	9.58		10.17	9.84	7.00	5.80	5.99	6.77	8.75	7.89		10.02	9.76	6.28	5.41	8.12	7.78	8.33	7.43
Time Saving Percentage	1.38	0.50	0.85	0.75	0.57	0.76	0.54	0.13		1.46	0.31	0.47	0.67	0.90	1.22	0.53	0.12		0.64	0.16	0.81	1.23	0.45	0.58	0.84	0.05

Figure 5. Experiment result of Big Buck Bunny.



	1.6 T0	1.6 T1	1.6 T2	1.6 T3	1.6 T4	1.6 T5	1.6 T6	1.6 T7		2.4 T0	2.4 T1	2.4 T2	2.4 T3	2.4 T4	2.4 T5	2.4 T6	2.4 T7		Dy T0	Dy T1	Dy T2	Dy T3	Dy T4	Dy T5	Dy T6	Dy T7
kW·h(Diskfull)	0.91	1.06	1.18	1.37	1.65	2.12	3.12	6.76		0.84	0.96	1.05	1.21	1.41	1.80	2.61	5.31		0.83	0.97	1.05	1.22	1.44	1.81	2.56	5.30
kW·h(Diskless)	0.84	0.94	1.09	1.24	1.49	1.91	2.81	6.12		0.77	0.85	0.99	1.13	1.35	1.67	2.35	4.94		0.76	0.89	0.99	1.12	1.33	1.66	2.39	4.93
Energy Saving Percentage	7.53	11.41	7.64	9.29	9.46	9.60	9.81	9.55		7.46	11.65	6.27	6.11	4.26	7.23	9.84	6.90		8.49	8.60	6.49	8.29	7.78	7.87	6.59	7.04
Time Saving Percentage	0.47	0.73	1.13	0.32	0.73	0.67	0.19	0.16		1.16	1.04	0.77	0.55	0.04	1.11	0.31	0.32		0.17	0.69	0.18	0.65	0.83	0.75	0.51	0.25

Figure 6. Experiment result of Elephant Dream.