The Computer Journal Advance Access published February 20, 2013

doi:10.1093/comjnl/bxt017

An Empirical Investigation of the Impact of Server Virtualization on Energy Efficiency for Green Data Center

YICHAO JIN[1,*], YONGGANG WEN[1], QINGHUA CHEN[2] AND ZUQING ZHU[3]

[1]*Nanyang Technological University, Singapore*
[2]*Yangtze Delta Institute of Tsinghua University, Zhejiang, PR China*
[3]*University of Science and Technology of China, Hefei, PR China*
[*]*Corresponding author: yjin3@ntu.edu.sg*

The swift adoption of cloud services is accelerating the deployment of data centers. These data centers are consuming a large amount of energy, which is expected to grow dramatically under the existing technological trends. Therefore, research efforts are in great need to architect green data centers with better energy efficiency. The most prominent approach is the consolidation enabled by virtualization. However, little effort has been paid to the potential overhead in energy usage and the throughput reduction for virtualized servers. Clear understanding of energy usage on virtualized servers lays out a solid foundation for green data-center architecture. This paper investigates how virtualization affects the energy usage in servers under different task loads, aiming to understand a fundamental trade-off between the energy saving from consolidation and the detrimental effects from virtualization. We adopt an empirical approach to measure the server energy usage with different configurations, including a benchmark case and two alternative hypervisors. Based on the collected data, we report a few findings on the impact of virtualization on server energy usage and their implications to green data-center architecture. We envision that these technical insights would bring tremendous value propositions to green data-center architecture and operations.

Keywords: virtualization; server consolidation; green data center; energy efficiency

Received 26 June 2012; revised 4 January 2013
Handling editor: Min Chen

1. INTRODUCTION

Recent advances in cloud computing are transforming the information and communication technologies (ICTs) [1]. The rapid adoption of cloud services is accelerating the deployment of data centers. These data centers in turn are contributing a dramatic growth of energy consumption in the ICT sector. It was estimated that data centers in the USA consumed ∼61 billion kWh of electricity in 2006, accounting for 1.5% of all US electricity consumption and more than doubled the energy consumption in 2000 [2]. The cost was expected to double again by reaching 120 billion kWh, or 4 kWh per person in 2011 [3]. Moreover, an annual growth rate at 30% on data-center energy consumption was predicted from 2012 to 2016 [4]. This exploding energy cost, which would overshadow the capital cost of data centers, must be tamed for a sustainable growth.

The energy usage in data centers consists of two parts, including energy consumed by the ICT subsystem (i.e. servers, storage and networking) and energy consumed by infrastructure (e.g. heating, ventilation, air-conditioning). This research focuses on the energy consumed by the ICT subsystem in data centers. Specifically, the energy usage by the ICT subsytem depends on both the specifications of individual building components and the operational configurations (e.g. applications and load balancing in the ICT subsystem) [5]. Previous research efforts have explored both areas of improvements for energy conservation. On the former aspect, early in 1992, the US Environmental Protection Agency introduced the ENERGY STAR as a voluntary labeling program to identify and promote energy-efficient products and to reduce greenhouse gas emissions and the electricity consumption of computers and monitors [6]. Recently, new hardware designs,

for example, next-generation memory solutions (i.e. DDR3 SRAM) and solid state drive, are emerging to improve both the computing speed and the energy efficiency in the ICT subsystem [7]. On the latter aspect, researchers aim to optimize the energy consumption of computing systems via dynamic resource allocation on an operation-system (OS) level and a data-center level [8, 9]. It has been generally accepted that server consolidation will improve the energy efficiency in data-center operations [10]. More specifically, it was expected [11] that savings on the order of 20% can be achieved in server and network energy consumption, by optimizing the data-center operations. Although some works have addressed the energy issue for data centers as a whole [12, 13], little effort has been paid to investigate the server energy usage under the context of cloud computing, in particular, the impact of data-center virtualization.

The technical feasibility of curbing the growing energy usage for the ICT subsystem in data centers is enabled by the latest development of cloud computing technologies. The most prominent solution is to consolidate applications from multiple servers to one server, enabled by server virtualization. On a first-order approximation, server virtualization could potentially reduce energy usage by turning off those idle savers. However, virtualization would also lead to other potential hazard effects, in particular, a possible overhead in energy usage and a possible reduction in maximum throughput. These detrimental effects, if not well understood and controlled, could offset the benefits of server virtualization. As a result, clear understanding and precise modeling of server energy usage in data centers, coupled with virtualization in cloud computing, will provide a fundamental basis for data-center operational optimizations, to accomplish the green ICT vision [14].

In this research, we investigate the impact of server virtualization on energy usage for data centers, with an ultimate goal to provide insights for optimizing data-center architecture and operations. In particular, we adopt an empirical approach to measure the energy consumed by servers under different virtualization configurations, including a benchmark case (i.e. physical machine) and two alternative hypervisors [i.e. Xen and Kernel-based virtual machine (KVM)]. Our experiments aim to characterize the energy overhead incurred by computing-intensive tasks and networking-intensive applications, corresponding to two important resources (computing and networking[1]) in cloud computing [16], under the situation in which a physical server is virtualized into multiple virtual machines (VMs). During the experiments, we obtain statistics for the CPU usage, the power level, the elapsed time and the energy consumption, under both local (internal) and network (external) traffic stresses.

[1]Energy consumption in storage is not investigated in this research, because its energy consumption is comparatively smaller than that for computing and networking [15].

Our empirical characterization generates fundamental understandings of server energy usage in the context of cloud computing and suggests engineering insights for energy-efficient data-center operations. In-depth analysis of the empirical results reveals a few fundamental insights about the impact of virtualization on server energy usage, including the following:

(1) Server architecture is still far from being energy-proportional in that a significant amount of power is consumed when the server is idle, thus opening an opportunity for server consolidation in data centers for reducing the energy cost.
(2) Virtualized servers consume more energy than physical ones, for both computing and networking-intensive traffic. The energy overhead from virtualized servers increases as the utilization of physical resources increases.
(3) The energy overhead resulting from server virtualization highly depends on the hypervisor used, which in turn is determined by the software architecture of the hypervisor (e.g. CPU scheduling and I/O design, etc.).
(4) From a given traffic load, the energy consumption can be minimized by launching an optimal number of VMs.
(5) In a multi-core server running multi-process applications, physical servers, if a multi-core optimization mechanism is absent, could consume more energy than virtualized servers.

These empirical insights suggest some operational optimizations toward an energy-efficient data center design and operations, including the following:

(1) Server consolidation for a green data center should aim to balance a fundamental trade-off between the energy saving from shutting down idle servers and the detrimental effects (i.e. the energy overhead and the throughput reduction from hypervisor) due to server virtualization.
(2) Hypervisors should be designed with energy consumption objectives, while providing the maximal flexibility in resource management.
(3) Resources should be allocated dynamically according to the real-time demand, with an objective to minimize the energy consumption.
(4) Multi-core scheduling algorithms should be incorporated in hypervisor design for virtualized servers and OS design for physical servers, to minimize the energy consumption.

The remainder of this paper is structured as follows. We describe the virtualization model and its impact on energy consumption in Section 2. Our detailed experimental setup is illustrated in Section 3. Section 4 presents the empirical results and the fundamental insights. Relevant engineering impacts on data-center operations are outlined in Section 5. In Section 6,

we explain the fundamental trade-off in server virtualization and its application to server consolidation in data-center operations. Section 7 presents previous work in similar realms. Section 8 concludes our work and discusses future research directions.

2. SERVER VIRTUALIZATION

One of the key technical enablers for server consolidation in cloud computing is virtualization. Specifically, a server administrator uses a software application (i.e. hypervisor) to divide one physical server into multiple isolated VMs, each of which runs its own guest operating system and specific applications. In this section, we present two leading virtualization models, including their implementation mechanisms in I/O, CPU and networking resource management. The impacts of these implementation details on server energy usage will be characterized with our proposed measurement objectives.

2.1. Virtualization model

Hypervisor, also referred to as VM manager, is one of hardware virtualization techniques that allow multiple operating systems to run concurrently on a host server. The hypervisor presents to the guest operating systems a *virtual operating platform* and manages the execution of the guest operating systems. Existing hypervisors, based on their relationship with the hardware platform, can be classified into two alternative types [17]:

(1) *Type* 1: hypervisor runs directly on the host's hardware to control the underlying hardware and to manage the guest operating system. The guest operating system thus runs on the level above the hypervisor.
(2) *Type* 2: hypervisor runs as a module within the operating system environment. In this case, the hypervisor layer can be considered as a second software level, while the guest operating system runs at the third level above the hardware.

In Fig. 1, we illustrate the logic structure of both types of hypervisors.

In this research, we focus on the leading open-source hypervisors, including Xen [18] and KVM [19], as exemplary hypervisors. Specifically, Xen is a type-1 hypervisor, which directly interfaces with the underlying hardware and uses a special, privileged domain 0 to manage other kernel modified guests [18]. KVM is designed as a type-2 hypervisor, in which the virtualization interface is designed to function the same as the actual physical hardware [19]. As such, their design principles and mechanisms are different in the areas of hardware resource scheduling, interrupt handling and VM management, as elaborated on the following subsections.

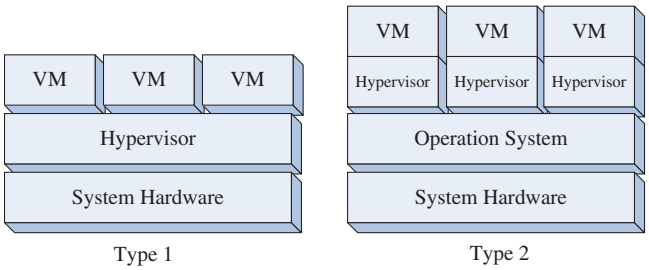

FIGURE 1. Two alternative types of hypervisors: type 1 hypervisor runs directly over the hardware, and type 2 hypervisor runs within the host system.

2.1.1. *Virtualized I/O mechanism*

Xen exposes a hypercall mechanism (also known as paravirtualization interface) in which all guest operating systems have to be modified to perform privileged operations (e.g. updating page table). Moreover, an event notification mechanism is proposed to deliver virtual interrupts derived from real device interrupts and other notifications to VMs.

KVM is an extension of the quick emulator (QEMU) emulator with support for the x86 VT extensions (VTx), and it typically uses full virtualization [19]. Guest operating systems above KVM do not need to change, and they appear as normal Linux processes instead of separated domains. When I/O instructions are issued by a guest operating system, a process context switch in the hypervisor is enabled to allow I/O signals passing through the hypervisor and the host OS.

The difference in virtualized I/O mechanisms for Xen and KVM directly impacts the energy consumption for virtualized servers. Specifically, Xen allows guest VMs to make system calls without invoking the kernel in the host OS, whereas KVM incurs additional kernel operations to support the I/O behaviors of guest systems. The set of additional operations translates to extra CPU cycles and memory access. Consequently, the KVM is expected consume more energy than Xen does, for similar traffic patterns and loads.

2.1.2. *Virtualized CPU model*

The current default CPU scheduler in Xen is a proportional fair share scheduler, called credit-based CPU scheduler [20]. This scheduler assigns each VM a weight, and an optional cap. The weight indicates the relative physical CPU allocation of a domain, and the cap sets an absolute limit by the percentage of a real CPU on the amount of time a domain can consume. The scheduler, running on a separate accounting thread in the host system, transforms the weight into a credit allocation for each virtualized CPU (VCPU). When a VCPU is running, it consumes the credit allocated to it. Once the VCPU runs out of the credit, it only runs when other, more thrifty VCPUs have finished their execution. Periodically, the accounting thread is reactivated and distributes more credits to VMs [21].

As a comparison, the KVM is a part of Linux and uses the regular Linux CPU scheduler and memory management [22]. It is known that by default, the KVM makes use of the Linux Kernel component, namely completely fair scheduler (CFS), to treat every KVM guest machine as a normal thread. Every task running on the KVM has a priority from 0 to 139 in the kernel. The range from 0 to 99 is reserved for real-time processes, and the range from 100 to 139 is for the user space. The smaller the priority number, the more important the task is viewed. In contrast to Xen's Credit Scheduler, the CFS scheduler implementation is not based on run queues. Instead, a red-black tree implements a timeline of future task execution. This data structure ensures that every runnable task chases other tasks to maintain a balance of execution across the set of runnable tasks including guest domains.

In spite of their different implementations, these two CPU schedulers fairly allocate CPU cycles among all the active guest systems. As a result, the CPU scheduler of either Xen or the KVM is capable of balancing global load on multi-processors to balance the both computing and energy usage, which will be verified by our measurements.

2.1.3. Virtualized networking model

Xen with the default configuration uses network bridging and a virtual firewall router (VFR) within domain 0 to allow all domains to appear on the network as individual hosts. Each guest domain attaches to one or more virtual network interfaces (VIFs), which consists of two I/O rings of buffer descriptors for transmitting and receiving, to identify its unique IP address.

In comparison, the KVM inherits the networking virtualization ability from QEMU to use TUN (network TUNnel)/TAP (network tap) in Linux kernel to create a virtual network bridge and routing. The bridge essentially emulates a software switch, allowing each VM to have individual networking resources.

The KVM would consume more energy than that consumed by Xen when they are exposed to networking-intensive tasks, because significant software operations are required by the KVM, while Xen takes advantage of its modified interface, which needs relatively less software participation.

2.2. Measurement objectives

The objective of our experiment is 2-fold. First, we aim to quantify and profile energy consumption of virtualized servers under different traffic patterns. Secondly, our measurement aims to map out the energy consumed in different virtualization components, as shown in Table 1. Our measurements will be based on comparison between virtualized servers and physical servers under local computing-intensive tasks and networking-intensive applications.

TABLE 1. Major components of virtualization model.

Structure	Xen Type 1 hypervisor	KVM Type 2 hypervisor
I/O mechanism	Hypercall and virtual event	Software contest switch and VTx
CPU scheduler	Credit-based scheduler	CFS
Networking	VFR and VIF	TUN and TAP

3. EXPERIMENTAL SETUP

This section describes our experimental environment.

3.1. Physical setup

Figure 2 illustrates our experimental setup, consisting of three identical physical servers. The machines under test are Inspur 3060 servers, each of which contains a quad-core Intel 2.13 GHz Xeon processor with 8 MB L2 cache, 2 GB RAM, 500 GB hard disk and a 1 Gigabit Ethernet card. All of them are connected to our test intranet over a D-link GDS-1024T 24-ports 1000 Base-T switch. We use CentOS 5.6-final-x86_64 with Linux kernel 2.6.18 as our OS platform for both host and guest systems. Xen 3.0.3 and KVM 83 are installed from CentOS packages on server machine B and machine C, respectively. Three guest VMs running on server B and C are allocated with 4 VCPUs, 512 MB RAM, 50 GB image and also installed CentOS 5.6-final-x86_64 as a guest OS. We leave all the software parameters intact. Besides, a Kill-A-Watt power meter, with a standard accuracy of 0.2%, is connected to each physical server, measuring the energy usage. Finally, our experiment is controlled by a desktop computer, connected to the testbed intranet as a monitor.

3.2. Test case design

We begin with collecting the background energy consumption when all the servers are idle. Following that, we launch a set of different profiles of local and network traffic to stress all three servers. During these tests, statistics on the CPU usage, the power consumption and the task completion time are collected. Detailed test cases are explained as follows.

3.2.1. Local computation benchmark

The local computational intensive task is simulated through accurately calculating π. In our experiment, *bc* command in Linux is utilized to calculate the mathematic constant π into an accurate level (100 000 digits after the decimal point). We simultaneously run multiple instances of this π-based CPU/RAM intensive application as concurrent processes to generate different local computing loads.

Specifically, five cases with an increasing number of concurrent processes, ranging from three to seven instances,

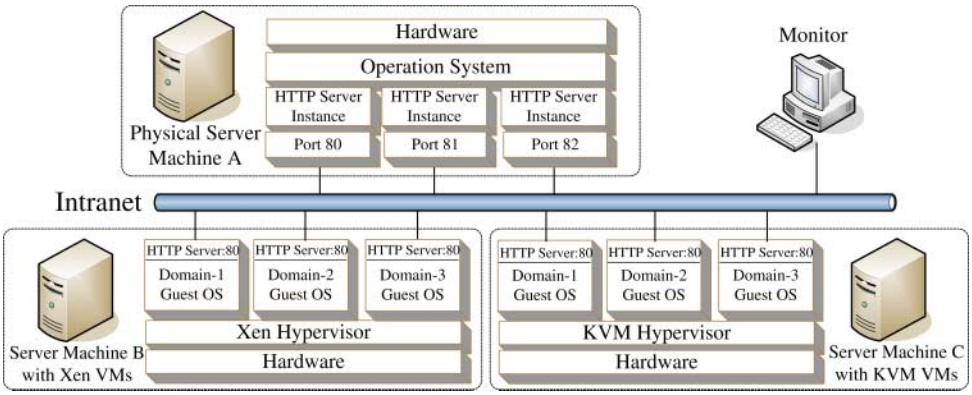

FIGURE 2. Experimental setup: three systems under test were configured, including a non-virtualized server, a Xen-virtualized server and a KVM-virtualized server.

are tested for two domain configurations (i.e. two or three active domains). On the physical machine, all the instances are executed on the same host OS, while on the virtualized servers, the concurrent instances are distributed evenly across all the active domains. For instance, when there are three guest domains running seven instances in parallel, two domains will execute two instances each and the last domain will execute three instances. The same configuration will rotate among all three guest domains and the final result will be calculated as the mean of three sets of data collected.

3.2.2. Http request benchmark

The network-intensive traffic benchmark is simulated through HTTP requests, since the HTTP-based traffic is the most widely used one, which accounts for more than 85% of the total web usage [23]. Our approach is similar to the web-server workload benchmark used in [24]. The configurations on the server and client side are explained as follows.

On the server side, three Apache servers (version 2.2.18) are configured on all servers under test. On the physical server (Server A), these three Apache servers are executed on three transmission control protocol (TCP) ports (80, 81 and 82) for traffic segregation. For virtualized servers (Server B and C), the three instances of HTTP servers are uniformly distributed across all active guest domains, for both domain configurations (i.e. two or three active domains). The same TCP ports are used for fair comparison. The contents stored on the HTTP servers are 1000 unique files retrieved from one commercial website, with a mean file size of 10.8 KB. In addition, we configure the Apache servers to allow a high density volume of web traffic, by increasing the *MaxClients*[2] value from 256 to 2000 and the *MaxRequestsPerChild*[3] value from 100 to 10 000.

On the client side, we use the *ab* (Apache Bench) tool [25] to simulate real web traffic. Specifically, three clients are configured to generate http GET requests at specific rates, each of which is towards one instance of the Apache server on one TCP port. Every client sends 5000 requests for each file, to cover two importance test cases of HTTP benchmark including: (i) single unique hit[4] and (ii) all unique hit[5]. Note that the cache miss test[6] is not conducted because we are only interested in networking energy consumption, which is being covered by the two cases tested. In this test profile, the overall size of data transferred is ∼150 GB, which is large enough to thoroughly exercise the network interface controller (NIC) device.

In our experiment, we gradually increase the request rate by the clients to scope the energy consumption as a function of the workload. Specifically, we set the request rate at 2500 reqs/s and 5000 reqs/s to simulate low web traffic workload, 10 000 reqs/s to simulate moderate workload, and 15 000 reqs/s to simulate peak workload, suggested by the workload for a real commercial web server [26].

3.3. Methodologies on result gathering

3.3.1. CPU utilization

We implement a Python script on each server to obtain the average CPU utilization during the test period. Specifically, by using the *top* command, it starts the monitor as soon as the workloads are initialized, and ends the data gathering once the workloads are completed. Finally, it can automatically record the obtained results into a local file.

[2] *MaxClients* sets the limit on the number of simultaneous requests that will be served.

[3] *MaxRequestsPerChild* sets the limit on the number of requests that an individual child server process will handle.

[4] For the single unique hit case, the same content file is served to the client consistently from the memory.

[5] In the all-unique test case, all content files are read from the disk into the memory to serve the client.

[6] In the cache-miss case, the content files are retrieved from remote servers to serve the client.

3.3.2. Elapsed time

For the local computation benchmark, the elapsed time is obtained by the server via a Python script. The script uses a timer to record the test period.

For the http request benchmark, the elapsed time is obtained by the monitor via a Python script. By using the *iftop* command, the script uses a timer to record the period when there are significant networking traffics.

3.3.3. Power/energy consumption

The average power consumption and the overall energy consumption is obtained by the readings on the power meters. Specifically, we manually reset the power meter before each test round, and get the reading including power and energy consumptions once each specific benchmark has been finished.

4. EMPIRICAL RESULTS AND FUNDAMENTAL INSIGHTS

In this section, we present our empirical findings of the impact of virtualization on server energy consumption. Moreover, we generalize a few fundamental insights from this set of empirical findings.

4.1. Background energy consumption

In Fig. 3, we plot the background power consumption of the physical machine A, the Xen-based server B and the KVM-based server C. Specifically, we measure the power consumed by all the servers when they are turned off and when they are turned on but idle. For the virtualized servers (i.e. B and C), two domain configurations (i.e. two or three active domains) are tested. The bar indicates the average power consumption, and the line refers to the fluctuation based on our observed maximum and minimum power during the test period.

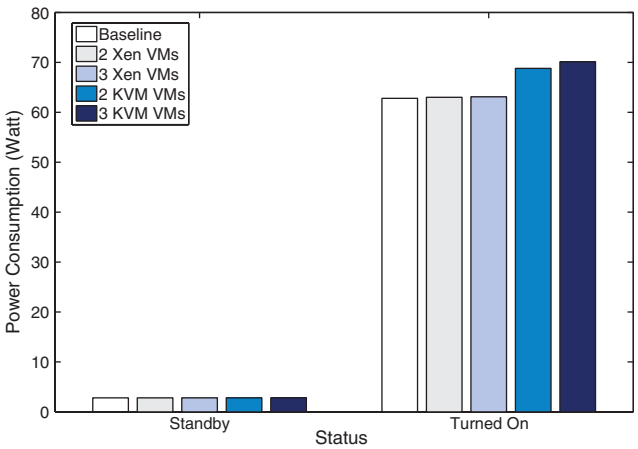

FIGURE 3. Background power consumption.

From Fig. 3, we have the following empirical findings:

Finding (a): All the servers consume about the same power (around 2.8 W) when turned off, but plugged into their power supplies. This amount of power is simply wasted in supporting server internal indicators (e.g. LEDs). Therefore, it is advantageous to turn off the power supply, instead of the servers, when servers are scheduled to shut down.

Finding (b): When servers are turned on and active VMs stay idle, the power overhead of different hypervisors against the physical server varies significantly from each other. Specifically, the Xen-based server consumes almost the same amount of power as the physical server does. Specifically, it consumes 63.1 W (three active Xen VMs) and 63.0 W (two active Xen VMs) of power, which is only 0.47 and 0.32% more than 62.8 W consumed by the physical server. However, the KVM-based server incurs a much higher overhead, consuming 70.1 W (11.6% overhead) for three active VMs and 68.8 W (9.55% overhead) for two active VMs. Moreover, the power usage of the KVM-based server C fluctuates within a wider range.

The Finding (b) can be explained by the different virtualization models adopted by Xen and the KVM, which impact the usage of CPU and RAM resources. The CPU utilization of the idle physical server is generally <0.3%, compared with ∼0.7–0.8% for the Xen-based server and 0.8–2.6% for the KVM-based server. The extra CPU usage of virtualized servers accounts for a portion of the energy overhead. The larger energy overhead for the KVM-based server can also be due to the larger memory footprint in the KVM, as indicated by the results of the memory test in [27, 28]. Tests are being conducted to collect additional data for principle component analysis to attribute the overhead into extra CPU cycle and memory footprint.

4.2. Local calculation benchmark

Results from the local computation benchmark are illustrated in Figs 4–8 and Table 2. Figures 4–7 present our measurements of the CPU usage, power consumption, task completion time and calculated energy consumption, respectively. Figure 8 presents the relative energy overhead consumed by the virtualized servers compared with the physical server. Table 2 summarizes the energy overhead for all the test cases.

In the following, a few key observations from these empirical results are explained.

Finding (c): It was observed that the virtualized server could consume less energy than the physical server does. Specifically, when five instances of the *bc* application are executed in parallel (the number of concurrencies is one more than CPU cores in the server), the energy overhead is negative for the Xen-based server as the valley point shown in Fig. 8.

Such an observation can be understood as the inter-play between the concurrent processes and the CPU cores in a multi-core server. For the physical server, we observe that four *bc* instances complete first and the last instance is completed much

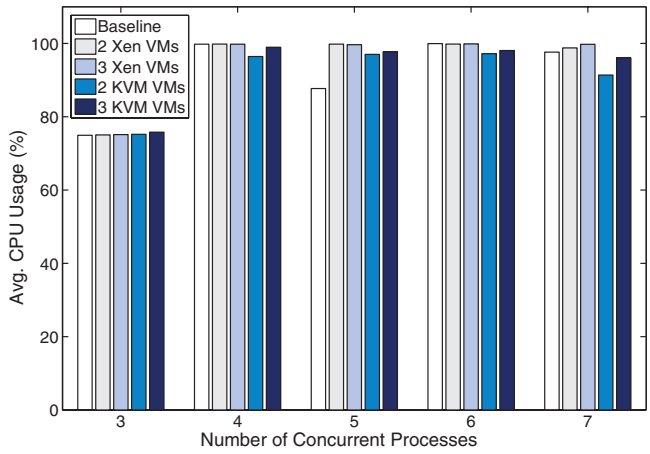

FIGURE 4. CPU usage comparison for local CPU-intensive task benchmark.

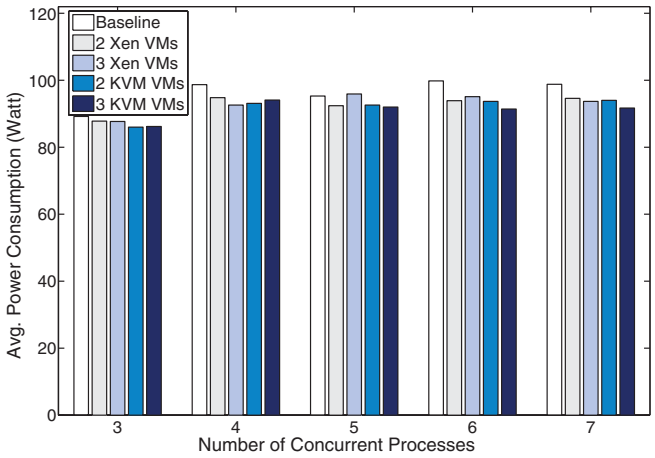

FIGURE 5. Power consumption comparison for local CPU-intensive task benchmark.

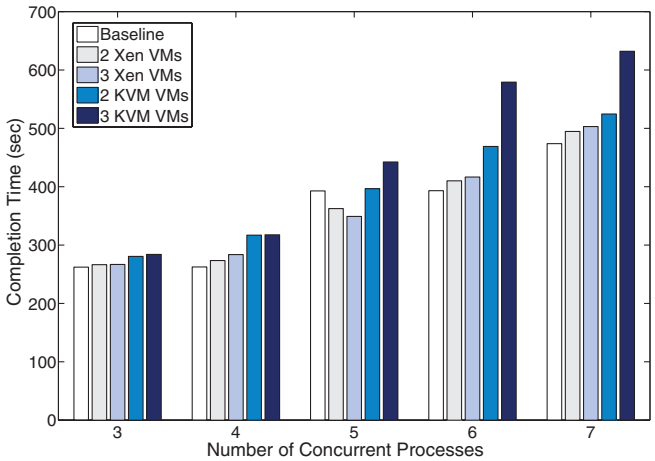

FIGURE 6. Completion time comparison for local CPU-intensive task benchmark.

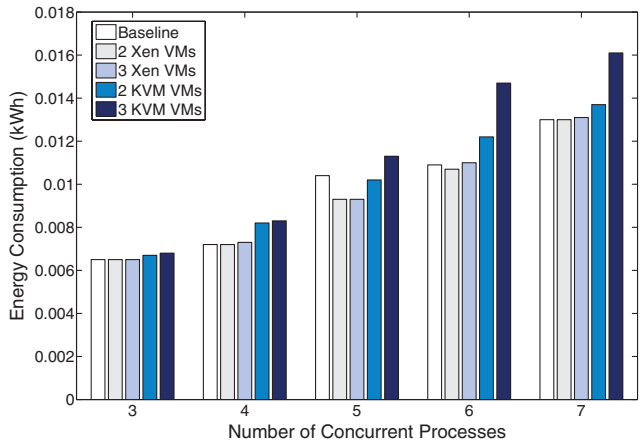

FIGURE 7. Energy consumption comparison for local CPU-intensive task benchmark.

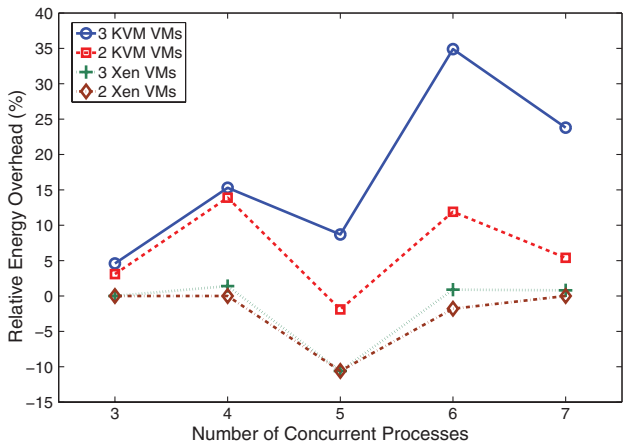

FIGURE 8. Energy overhead of CPU-intensive benchmark: the abnormality for five concurrent processes is explained in Finding (c).

later. This is further verified by the observation that the CPU usage maintains nearly 100% until the first four instances are completed, and then it drops to around 25% afterward. This observation can be contributed to a lack of multi-core scheduling mechanism in Linux across different processes, which can be compensated via a multi-thread processing paradigm. As a comparison, in the virtualized servers, all the instances complete almost at the same time, due to the built-in CPU scheduler in hypervisors allocating CPU cycles across active VMs. Consequently, the CPU usage on the Xen-based server maintains at a high level of 99.8%, compared with 87.7% for the physical server. In this case, the Xen-based server, either running two or three VMs, takes ∼10% less time and consumes 11% less energy than that of the physical server. For the KVM-based server, the advantage of the CPU scheduler is reversed by the extra penalty of hypervisor in most cases, except for the case when two active VMs are configured for the test case, resulting in a saving of 2% energy compared with the cost of

TABLE 2. Energy consumed by CPU-intensive benchmark (unit: kWh).

Workload	Three tasks	Four tasks	Five tasks	Six tasks	Seven tasks
Baseline	0.0065	0.0072	0.0104	0.0109	0.0130
Overhead	0.0%	0.0%	0.0%	0.0%	0.0%
Two Xen	0.0065	0.0072	0.0093	0.0107	0.0130
Overhead	0.0%	0.0%	−10.6%	−1.8%	0.0%
Three Xen	0.0065	0.0073	0.0093	0.0110	0.0131
Overhead	0.0%	1.4%	−10.6%	0.9%	0.8%
Two KVM	0.0067	0.0082	0.0102	0.0122	0.0137
Overhead	3.1%	13.9%	−1.9%	11.9%	5.4%
Three KVM	0.0068	0.0083	0.0113	0.0147	0.0161
Overhead	4.6%	15.3%	8.7%	34.9%	23.8%

the physical server. This observation suggests that if there is no binding between running processes and CPU-cores, native OS cannot truly take advantage of multi-core architecture; in contrast, virtualized systems, based on either Xen or the KVM, is able to partition computing resources into smaller pieces to achieve energy saving.

Finding (d): The KVM-based server consumes more energy than the Xen-based server. For example, when processing seven concurrent tasks, two KVM VM-based server consumes 5.4% energy more than that based on two Xen VMs and the gap reaches 23% between three KVM VMs and three Xen VMs. This is due to the fact that the KVM hypervisor consumes more CPU cycles and occupies a higher memory footprint, compared with the Xen hypervisor. The additional requirement for system resources translates into higher energy consumption.

Finding (e): The number of active VMs affects the energy consumption for the KVM-based server. Specifically, when configuring three active VMs, the KVM-based server consumes more energy than that consumed by two active VMs configured on the same server. Such an observation can be understood from the frequent lock holder preemption (LHP) mechanism, investigated in [22]. A guest VCPU in the KVM-based server is a normal thread in the host operating system, which may be preempted when the host system de-schedules the VCPU threads. If the preempted VCPU is running in a critical section, the lock will be held for a long time from the perspective of the guest operating system. The probability of LHP increases with a higher number of active VMs. As a result, CPU resources are simply wastes in the lock holding period, which in turn increases the task completion time. It is observed that the average power consumption for the KVM-based server with three active VMs is the lowest, but the task completion time is the longest, resulting in a high energy consumption. This suggests that the number of VMs can be optimally configured to reduce the energy consumption for the KVM-based server.

4.3. HTTP request benchmark

Results from the HTTP request benchmark are plotted in Figs 9–13 and Table 3, respectively. Figures 9–12 present the statistics collected for our HTTP request benchmark test, including the average CPU usage, average power, task completion time and total energy consumption. Figure 13 illustrates the energy overhead for the virtualized servers, compared with the physical server. Table 3 summarizes the result of the energy consumption for different test cases.

In the following, we highlight a few findings suggested by the set of data collected for the HTTP request test.

Finding (f): The virtualization overhead for network-intensive traffic is significantly larger than that for computing-intensive traffic. For the Xen-based server, the energy overhead for computing-intensive traffic is <5%, while the overhead for network-intensive traffic could rise up to 70%. The same situation happens to the KVM-based server.

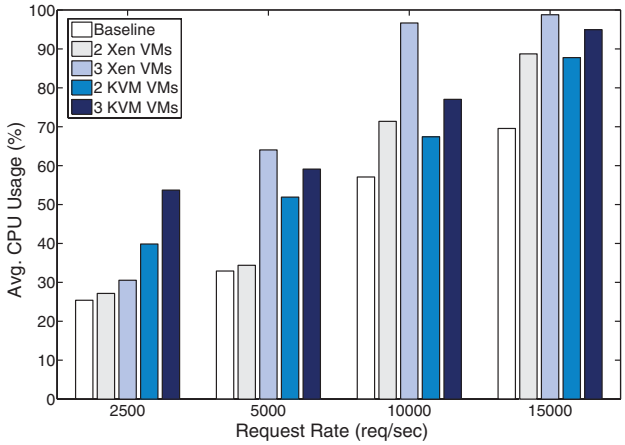

FIGURE 9. CPU usage comparison for the HTTP benchmark.

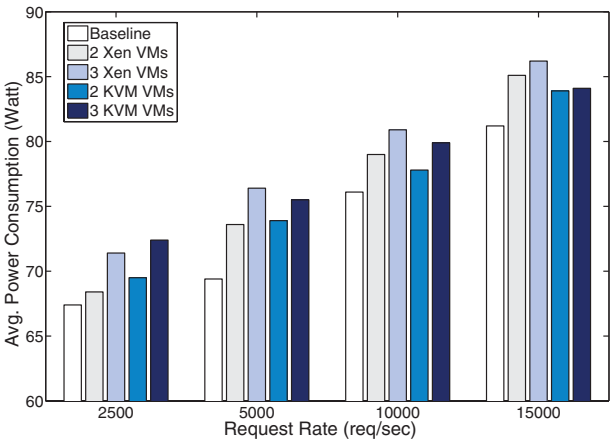

FIGURE 10. Power consumption comparison for the HTTP benchmark.

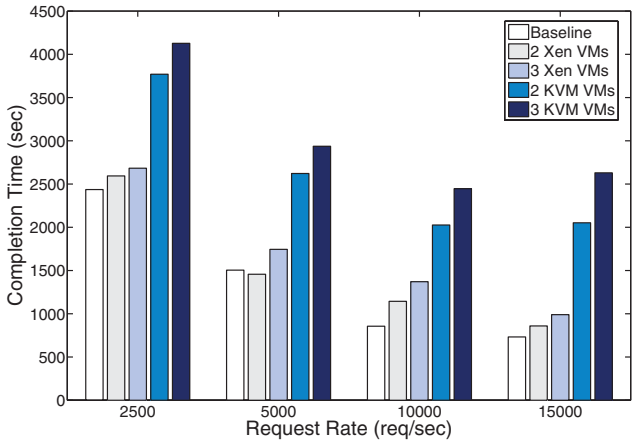

FIGURE 11. Completion time comparison for the HTTP benchmark.

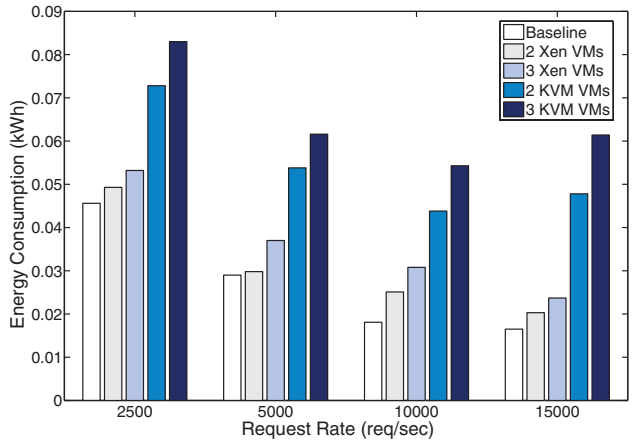

FIGURE 12. Energy consumption comparison for the HTTP benchmark.

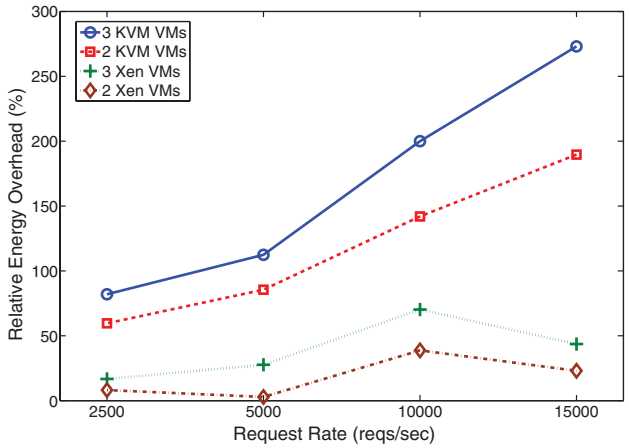

FIGURE 13. Relative energy overhead of the networking benchmark.

TABLE 3. Energy consumed by the networking benchmark (unit: kWh).

Workload	2500 req/s	5000 req/s	10 000 req/s	15 000 req/s
Baseline	0.0456	0.0290	0.0181	0.0165
Overhead	0.0%	0.0%	0.0%	0.0%
Two Xen	0.0493	0.0298	0.0251	0.0203
Overhead	8.1%	2.8%	38.7%	23.0%
Three Xen	0.0532	0.0370	0.0308	0.0237
Overhead	16.7%	27.6%	70.2%	43.6%
Two KVM	0.0728	0.0538	0.0438	0.0478
Overhead	59.6%	85.5%	142.0%	189.7%
Three KVM	0.0830	0.0616	0.0543	0.0614
Overhead	82.0%	112.4%	200.0%	273.1%

The cause of this finding is at least 2-fold. First, note that, for network-intensive traffic, the CPU usage for virtualized servers is higher than that for the physical server; while for computing-intensive traffic, the CPU usage for all the servers is almost equal for the same test case. This difference suggests that a significant amount of CPU usage is budget for the hypervisors to handle the vNIC operations (i.e. VFR/VIF for Xen and TUN/TAP for KVM). Secondly, it is shown in [29] that the probability of the occurrence of LHP for I/O-intensive workloads reaches 39% on average for virtualized machine. As a result, it takes much longer to complete the task by a virtualized system, translating into higher energy cost.

Finding (g): The energy overhead for the virtualized server is highly correlated with the number of active VMs. Specifically, the energy overhead is consistently higher for a larger number of active VMs. The energy overhead for three active VMs in the KVM-based server is around 1.5 times higher than that for two active VMs; similarly, the energy overhead for three active VMs in the Xen-based server is almost twice of that for the case of two active VMs. Moreover, the gap for the KVM-based server is higher. For example, in the case of 15 000 req/s, the overhead gap between three active VMs and two active VMs for the KVM-based server is more than 80%; while it is around 20% for the Xen-based server. The explanation could be that more active guests aggravate the impact of LHP. Therefore, it takes much longer to complete the task with more active guest domains, resulting in additional energy consumption.

Finding (h): The network throughput for the KVM-based server reaches its maximum between 10 000 and 15 000 reqs/s, while the network throughput for the physical server and the Xen-based server continue to grow up as the traffic rate increases. This observation can be made clearer in Figs 14 and 15, where the task completion time and the energy cost are plotted as a function of the request rate. Specifically, when the request rate is 15 000 reqs/s, the KVM-based server takes longer time to complete the task and thus consumes more energy, compared with the case of 10 000 reqs/s; as a comparison, the task completion time and the energy cost for the physical

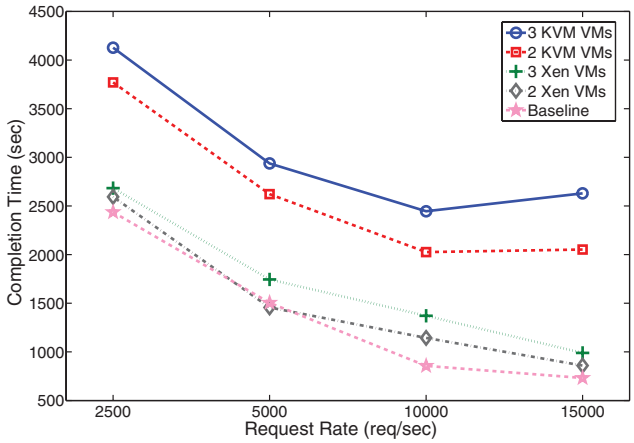

FIGURE 14. Completion time curve for the HTTP benchmark.

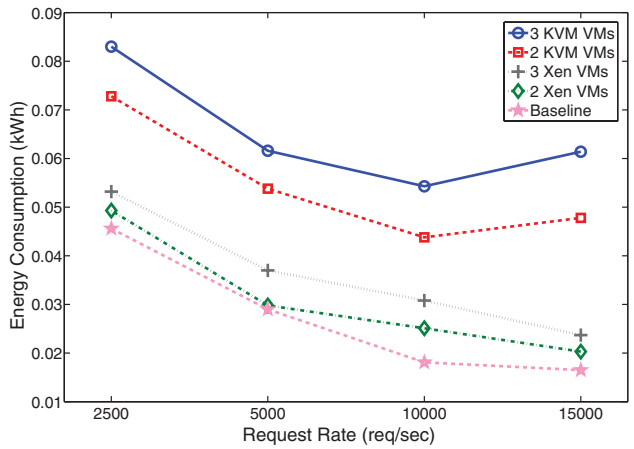

FIGURE 15. Energy consumption curve for the HTTP benchmark.

machine and the Xen-based server monotonically decrease as the request rate increases up to 15 000 reqs/s.

This observation is largely due to the extra memory footprint for the KVM hypervisor. In the Apache web server, each serving request takes some amount of memory. The maximum number of requests that can be served simultaneously is thus proportional to the amount of available resources. For the case of the KVM hypervisor, the extra memory footprint shrinks the amount of available memory for request serving. As a result, the network throughput is smaller compared with the physical server and the Xen-based server.

Finding (i): The marginal power consumed by the server under different load conditions is limited, compared with the power consumed when the server is idle. Specifically, the additional power consumed by the server under different levels of networking requests is at most 37.3% against the idle state, and the maximum additional power consumption for the local computation benchmark is 57.6%. Moreover, the marginal power consumption is highly correlated with the CPU usage

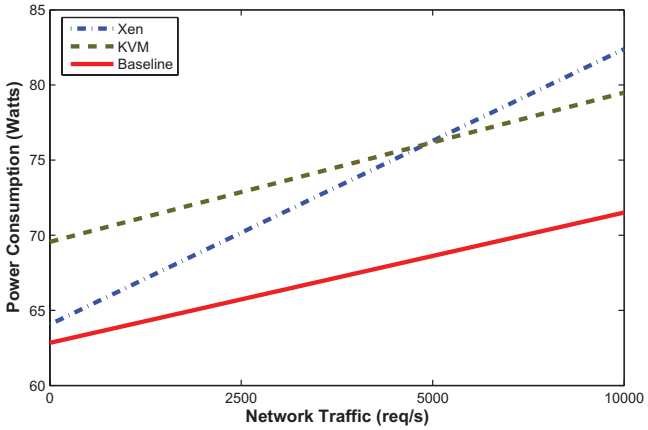

FIGURE 16. Expansive energy overhead due to server virtualization.

observed in our experience. As a result, our experiment verifies a previous power consumption model for the server, in which the power consumption of the server can be viewed almost as an affine function of CPU usage with the idle power consumption as the y-intercept [30]. It is desirable for the y-intercept to be as small as possible, to achieve an energy-proportional architecture.

Finding (j): The energy overhead for the virtualized servers is expansive, as a function of the traffic throughput. As illustrated in Fig. 16 where the lines are generated by first degree polynomial curve fitting based on the power consumption of different configurations, the power gap between the baseline and the virtualized servers for both Xen and KVM hypervisors increases as the throughput increases, before the maximum throughput of the KVM-based server is reached. When there is no network traffic, the gap between Xen and baseline is around 1% (0.8 W), and the gap between the KVM-based server and the baseline server is ∼10% (6.9 W). When the throughput grows to 10 000 reqs/s, the gap becomes 15.2% (10.8 W) for Xen and 11.2% (7.9 W) for the KVM.

4.4. Fundamental insights

Generalizing from the list of empirical findings, we present the following fundamental insights about the impact of virtualization on server energy consumption, as follows:

(1) The server is still far from energy-proportional. The idle server even consumes approximately two-thirds of the energy when its computing resource is fully occupied. Hence, it will be advantageous to consolidate applications from multiple servers to one and turn off those idle servers to save energy.
(2) The virtualized server in general consumes more energy than the physical server does, for both computing-intensive and networking-intensive tasks, even when they are idle. Besides, the energy overhead for the

virtualized servers increases as the resource utilization increases. When the virtualized servers are idle, the Xen hypervisor incurs <1% energy overhead, and the KVM hypervisor contributes around 10% extra energy cost. For the networking-intensive benchmark, Xen's VFR, VIF and virtual event mechanism add an energy overhead ranging from 2.8 to 70.2% for different workloads and VM configurations; and the KVM's Linux kernel-based virtual network bridge and x86 VTx invoking results in an energy overhead between 59.6 and 273.1% for various combinations of configuration and workload.

(3) The two types of hypervisors exhibit different characteristics in energy cost for various tasks. The KVM-based server consumes more energy than the Xen-based server under the same test case, due to their different architectural principles. Specifically, the KVM embeds the virtualization capability into the native Linux kernel. As a result, it adds one more layer into the software stack and accordingly consumes more energy. Moreover, KVM's networking model consumes more energy (on average nearly twice) than that of Xen, and the KVM-based server consumes on average 12% more energy than its counterpart with a XEN hypervisor. On the other hand, the benefit of using a type-2 hypervisor is that it requires almost no modification to the host operating system; while the host operating system needs to be modified to work with a type-1 hypervisor.

(4) Significant energy saving can be achieved by launching the traffic to an optimum number of VMs. In our measurement, the virtualized server with two active VMs consumes less energy than the one with three active VMs. Specifically, ∼20% energy for the KVM and 15% energy for Xen on average could be conserved for all cases under the network-intensive traffic benchmark, by migrating tasks from one VM to another and turning off the idle VM. For the computing-intensive traffic benchmark, a similar pattern is observed for the KVM-based server when more than five *bc* instances are executed in parallel.

(5) When a multi-core server is running multi-process applications, the physical machine could consume more energy than the virtualized servers. It is due to a lack of the multi-core optimization in the physical machine. By default, each application is executed with a single thread, which can be processed only by one CPU core, even if other cores are idle. As a comparison, both Xen and the KVM are able to distribute physical CPU cores into virtual CPU cores, avoiding starvation. This demonstrates one key advantage of virtualization in improving resource utilization.

To connect these insights with our empirical findings, we summarize them in a tabular format in Table 4. In the table, an entry marked with 'M' indicates that the corresponding finding in that column supports the insight associated with the corresponding row.

TABLE 4. Match-up between findings and insights.

	O								
I	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)
(1)	M		M					M	
(2)	M								M
(3)	M		M				M		
(4)				M		M			
(5)		M							

'M' marks the supporting relationship.

5. ENGINEERING IMPLICATIONS

As mentioned previously, the purpose of our empirical study is to develop engineering guidelines for energy-efficient architecture and operations of green data centers. In this section, we leverage the empirical findings and the derived fundamental insights to suggest some engineering implications for data-center operations.

5.1. Hypervisor optimization

Our measurements indicate that both hypervisors incur an energy overhead, compared with the physical server. This suggests that, in addition to the standard performance optimization for hypervisors, minimizing energy consumption should be another objective in their software architecture.

Xen, as a type 1 hypervisor, is more energy efficient than the KVM under both networking and local computation traffic benchmarks. In some case, Xen's CPU scheduler even makes it more energy efficient than the physical machine for the multi-process application. However, the energy cost for networking-intensive traffic for the Xen-based server is still relatively expensive. Some studies [20, 31] have aimed to optimize the mechanism of Xen to improve the performance, such as tuning the weight and cap value in Xen domain 0, optimizing the CPU scheduler algorithm and networking structure like netfilter on bridge, etc. Those efforts are provable to improve Xen's performance, but still need verifications when performance and energy consumption are both concerned.

The KVM runs inside the Linux kernel, making it easier to deploy and manage than Xen. However, there is always a trade-off between convenience and efficiency. Based on our observations, it spends more energy on coordinating all the guest OSes, even when they are idle. As such, some kind of energy-efficient CPU scheduler could be proposed for the KVM. In addition, strategies toward LHP avoidance should be in order.

Finally, given the advantageous energy performance of the Xen para-virtualization established in our study, introducing a virtual hardware mechanism could improve the energy efficiency for the KVM.

5.2. Dynamic resource management

Our study suggests a two-tier dynamic resource management framework for data-center operations. First, in a more coarse granularity, physical servers can be turned off, completely saving the energy cost, if the total load is not high enough. This optimization is derived from the fact that the idle server consumes a large portion of power compared with its fully loaded situation. Secondly, in a more fine granularity, VMs can be shut down and/or migrate into other physical server to save energy. This optimization is derived from our observation that energy consumption is sensitive to the number of active VMs and is enabled by the live migration mechanism in virtualization [32].

5.3. Multi-thread programming paradigm for multi-core system

Modern servers are usually equipped with multiple CPU cores. It provides an opportunity to enhance the raw computing capability; however, to harness such a raw capability additional cares are required for programming practice when applications are deployed onto data centers. As shown in our measurement, the energy consumption is sensitive to the binding between applications and CPU cores. As a result, the software architect should decide whether to bind applications with CPU cores judiciously for the non-virtualized servers (or adopt a multi-thread programming paradigm); while the hypervisor inherently provides the capability to distribute the computing capability among different VMs.

6. FUNDAMENTAL TRADE-OFF IN SERVER VIRTUALIZATION

Our research has also revealed a fundamental trade-off for server virtualization, as illustrated in Fig. 17.

On one hand, the energy consumption in data centers can be reduced by consolidating applications from multiple servers to one server and shutting down the idle servers. This insight can be obtained from our observation of an affine power consumption model for native servers.

On the other hand, for the virtualized servers, there are two detrimental effects that would hamper their energy efficiency. First, the hypervisor introduces a potential energy overhead over the physical machine, by allocating system resources for its execution. This overhead is expansive as a function of the 'goodput', which denotes the portion of computation capabilities used for support applications. Secondly, the maximum supportable goodput for the virtualized server is

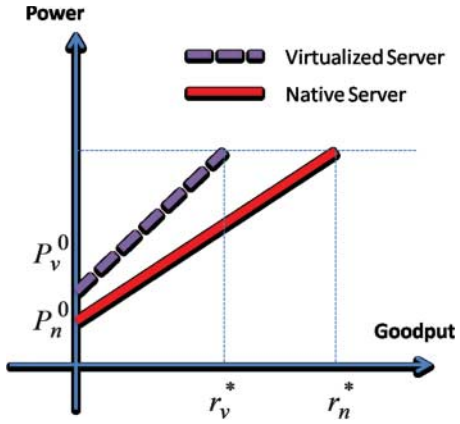

FIGURE 17. Fundamental trade-off in server virtualization.

reduced, compared with its native server. The combination of these two detrimental effects would offset the energy benefit of server consolidation. Moreover, the impact of these detrimental effects depends on the type of hypervisor chosen. Specifically, the relationship between the goodput and the power consumption for the native server can be expressed as

$$P_{\mathrm{n}} = P_{\mathrm{n}}^0 + (P_{\max} - P_{\mathrm{n}}^0) * r_{\mathrm{n}}/r_{\mathrm{n}}^{\star}, \tag{1}$$

where P_{n} is the power consumption for the native server, P_{n}^0 is the power consumption when the native server is idle, $P_{\max}$ is the maximum power consumption, r_{n} is the goodput and $r_{\mathrm{n}}^{\star}$ is the maximum supportable goodput.

Similarly, for the virtualized server, the relationship is as follows:

$$P_{\mathrm{v}} = P_{\mathrm{v}}^0 + (P_{\max} - P_{\mathrm{v}}^0) * r_{\mathrm{v}}/r_{\mathrm{v}}^{\star}, \tag{2}$$

where $P_{\mathrm{v}}^0 > P_{\mathrm{n}}^0$ and $r_{\mathrm{v}}^{\star} < r_{\mathrm{n}}^{\star}$. The detailed value of P_{v}^0 and $r_{\mathrm{v}}^{\star}$ largely depends on the type of hypervisor.

This fundamental trade-off dictates how server consolidation should be designed to reduce energy usage for green data centers. Specifically, the decision of server consolidation should balance the two competing forces, to reduce the energy usage by data centers.

7. RELATED WORK

The popularity of open-source hypervisors (i.e. Xen and KVM) benefits from an easy access to their detailed design. In [18], Xen was described as an x86 VM monitor that allows multiple commodity operating systems to share conventional hardware in a safe and resource-managed fashion without sacrificing either performance or functionality. Alternatively, the KVM emerges as a new Linux subsystem that leverages virtualization extensions to add a VM monitor (or hypervisor) capability to Linux [19].

Deshane *et al.* presented the initial quantitative analysis of these two hypervisors in [33], focusing on the overall

performance, performance isolation and scalability of VMs running on them. An extensive empirical performance measurement on such evaluation was conducted in [27, 28]. In [22], the authors analyzed the CFS CPU scheduler of the KVM on a multi-core environment, and proposed optimizations for performance enhancement. Huang *et al.* [34] presented a framework for VM-based computing for high performance computing applications. It built a computing cluster with VMs, and evaluates the performance as well as the cost compared with a native, non-virtualized environment.

Kim *et al.* [35] investigated a power-aware provisioning of VMs for real-time services, with a dual objective of reducing operating costs and improving system reliability. The architectural principles for energy-efficient management of clouds, resource allocation policies and scheduling algorithms were discussed in [36], demonstrating the immense potential of cloud computing to offer significant cost savings under dynamic workload scenarios. Berl *et al.* [37] reviewed the methods and technologies currently used for energy-efficient operation of computer hardware and network infrastructure. It also concluded that cloud computing with virtualization can greatly reduce the energy consumption.

Another area of study in the green ICT focuses on attributing energy cost to various components in a computing system. Agarwal *et al.* [38] characterized the energy consumed by desktop computers, suggesting that one typical desktop PC consumes 80–110 W when active, and 60–80 W when idle. Bohrer *et al.* [26] created a power simulator for web-serving workloads that is able to estimate CPU energy consumption with a low error rate. Contreras and Martonosi [39] used hardware event counters to derive power estimates that are accurate at sub-second time intervals. Our approach is similar to that of Economou *et al.* [15], which focuses on the coarser activity metrics such as the CPU load and I/O activities by using a variety of workloads. The novelty of our research is that we are the first group to cast this research agenda in a data center under the cloud computing context. The results obtained in our research shed new lights in energy-efficient architecture and operations of data centers, which in turn would help to curb the energy explosion predicted for ICT systems.

8. CONCLUSION AND FUTURE WORK

This paper reported an empirical study on the impact of server virtualization on energy efficiency for green data centers. Through intensive measurements, we have obtained sufficient statistics for energy usage for both the native server and the virtualized servers with two alternative hypervisors (i.e. Xen and KVM). Through an in-depth analysis of this dataset, we presented a few important findings, regarding the energy usage of the virtualized server, and crucial implications of these empirical findings. Moreover, our investigation suggested a few engineering implications for architecture green data centers. Finally, the most important result from our study is the fundamental trade-off in virtualized servers, which would dictate how server consolidation should be designed and deployed to tame the explosive energy usage in data centers.

Our future work will concentrate on three domains, including measurements, model and verifications. First, a green modular data center, which consists of more than 270 physical servers, is under construction at Nanyang Technological University. We will continue to measure the energy usage with the virtualized servers under a real web trace with combined traffic features (including HTTP traffic, P2P traffic, VoIP, etc.) in this data center. Secondly, based on our fine-granulated measurements, we will develop an analytical model for energy usage of the virtualized servers and focus on optimization techniques for server consolidation in green data-center operations. Finally, our optimization strategies will be configured on our data center and additional measurements will be taken to verify their effectiveness. The ultimate goal of this research is to develop an analytical framework for green data centers and provide a list of best practices for energy-efficient operations for data centers.

ACKNOWLEDGEMENTS

This research benefited from the discussions with a few world-class researchers in the area of green ICT research. Dr Dan Kilper and Dr Kyle Guan, from Bell Laboratories, were inspirational to our research and helped to make sure that our research was on the right track. Our research also benefited from the discussions with Mr Keok Tong LING at IDA, Singapore.

FUNDING

This research is supported by "Green Data Centre Innovation Challenge" funding scheme from "Infocomm Development Authority of Singapore".

REFERENCES

[1] Armbrust, M. *et al.* (2010) A view of cloud computing. *Commun. ACM*, **53**, 50–58.

[2] Brown, R. *et al.* (2008) Report to Congress on Server and Data Center Energy Efficiency: Public Law 109–431. Technical Report, Berkeley, CA, USA.

[3] Le, K., Bianchini, R., Nguyen, T., Bilgir, O. and Martonosi, M. (2010) Capping the Brown Energy Consumption of Internet Services at Low Cost. *Int. Green Computing Conf.*, Chicago, USA, pp. 3–14. IEEE, Washington.

[4] Corcoran, P. (2012) Cloud computing and consumer electronics: a perfect match or a hidden storm. *IEEE Consum. Electron. Mag.*, **1**, 14–19.

[5] Beloglazov, A., Buyya, R., Lee, Y., Zomaya, A. (2011) A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Adv. Comput.*, **82**, 47–111.

[6] Energy star. http://www.energystar.gov/.

[7] Samsung. Samsung green memory. http://www.samsung.com/greenmemory.

[8] Wierman, A., Andrew, L. and Tang, A. (2009) Power-Aware Speed Scaling in Processor Sharing Systems. *IEEE INFOCOM'09*, Rio, Brazil, pp. 2007–2015. IEEE, Washington.

[9] Meisner, D., Gold, B. and Wenisch, T. (2009) PowerNap: eliminating server idle power. *ACM Sigplan Not.*, **44**, 205–216.

[10] Hewitt, C. (2008) Orgs for scalable, robust, privacy-friendly client cloud computing. *IEEE Internet Comput.*, **12**, 96–99.

[11] Accenture (2008) Data Centre Energy Forecast Report-Final Report. Technical Report, Oakland, CA, USA.

[12] Benini, L., Bogliolo, A. and De Micheli, G. (2000) A survey of design techniques for system-level dynamic power management. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, **8**, 299–316.

[13] Gandhi, A., Harchol-Balter, M., Das, R. and Lefurgy, C. (2009) Optimal Power Allocation in Server Farms. *ACM SIGMETRICS'09*, Seattle, USA, pp. 157–168. ACM.

[14] Greentouch. http://www.greentouch.org/.

[15] Economou, D., Rivoire, S., Kozyrakis, C. and Ranganathan, P. (2006) Full-System Power Analysis and Modeling for Server Environments. *Workshop on Modeling, Benchmarking, and Simulation*, Boston, USA, pp. 70–77. IEEE, Washington.

[16] Mell, P. and Grance, T. (2011) *The NIST definition of cloud computing (draft)*. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf.

[17] IBM. *IBM system virtualization version 2 release 1*. http://publib.boulder.ibm.com/infocenter/eserver/v1r2/topic/eicay/eicay.pdf.

[18] Barham, P. *et al.* (2003) Xen and the Art of Virtualization. *ACM SIGOPS Operating Systems Review*, New York, USA, pp. 164–177. ACM.

[19] Kivity, A., Kamay, Y., Laor, D., Lublin, U. and Liguori, A. (2007) KVM: The Linux Virtual Machine Monitor. *Linux Symp.*, Ottawa, Canada, pp. 225–230. http://www.linuxsymposium.org/2007/cfp.php.

[20] Cherkasova, L., Gupta, D. and Vahdat, A. (2007) Comparison of the three CPU schedulers in Xen. *ACM SIGMETRICS Perform. Eval. Rev.*, **35**, 42–51.

[21] *Xen credit-based CPU scheduler*. http://wiki.xensource.com/xenwiki/CreditScheduler.

[22] Jiang, W., Zhou, Y., Cui, Y., Feng, W., Chen, Y., Shi, Y. and Wu, Q. (2009) CFS Optimizations to KVM Threads on Multi-Core Environment. *Int. Conf. Parallel and Distributed Systems (ICPADS'09)*, Shenzhen, China, pp. 348–354. IEEE, Washington.

[23] Schulze, H. and Mochalski, K. *Internet study 2008/2009*. http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf.

[24] Menon, A., Santos, J., Turner, Y., Janakiraman, G. and Zwaenepoel, W. (2005) Diagnosing Performance Overheads in the Xen Virtual Machine Environment. *Int. Conf. Virtual Execution Environments*, Chicago, USA, pp. 13–23. ACM.

[25] ApacheBench. http://en.wikipedia.org/wiki/ApacheBench.

[26] Bohrer, P. *et al.* (2002) Power Aware Computing, Chapter. *The Case for Power Management in Web Servers*, pp. 261–289. Kluwer Academic Publishers, Norwell, MA, USA.

[27] Che, J., He, Q., Gao, Q. and Huang, D. (2008) Performance Measuring and Comparing of Virtual Machine Monitors. *IEEE/IFIP Int. Conf. Embedded and Ubiquitous Computing (EUC'08)*, Shanghai, China, pp. 381–386. IEEE, Washington.

[28] Che, J., Yu, Y., Shi, C. and Lin, W. (2010) A Synthetical Performance Evaluation of OpenVZ, Xen and KVM. *IEEE Asia-Pacific Services Computing Conf. (APSCC'10)*, Hangzhou, China, pp. 587–594. IEEE, Washington.

[29] Uhlig, V., LeVasseur, J., Skoglund, E. and Dannowski, U. (2004) Towards Scalable Multiprocessor Virtual Machines. *Virtual Machine Research and Technology Symp.*, San Jose, USA, pp. 43–56. USENIX, Berkeley.

[30] Fan, X., Weber, W. and Barroso, L. (2007) Power provisioning for a warehouse-sized computer. *ACM SIGARCH Comput. Archit. News*, **35**, 13–23.

[31] Santos, J., Janakiraman, G., Turner, Y. and Pratt, I. (2007) *Netchannel 2: optimizing network performance, Xen-Source/Citrix Xen Summit*. http://www.xen.org/files/xensummit_fall07/16_JoseRenatoSantos.

[32] Verma, A., Ahuja, P. and Neogi, A. (2008) pMapper: Power and Migration Cost Aware Application Placement in Virtualized Systems. *Int. Conf. Middleware*, Leuven, Belgium, pp. 243–264. Springer, New York.

[33] Deshane, T. *et al.* (2008) *Quantitative comparison of Xen and KVM, Xen summit*. http://www.xen.org/files/xensummitboston08/Deshane-XenSummit08-Slides.

[34] Huang, W., Liu, J., Abali, B. and Panda, D. (2006) A Case for High Performance Computing with Virtual Machines. *Int. Conf. Supercomputing*, Queensland, Australia, pp. 125–134. ACM.

[35] Kim, K., Beloglazov, A. and Buyya, R. (2011) Power-aware provisioning of virtual machines for real-time cloud services. *Concurrency Comput. Pract. Exp.*, **23**, 1491–1505.

[36] Beloglazov, A., Abawajy, J. and Buyya, R. (2012) Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future Gener. Comput. Syst.*, **28**, 755–768.

[37] Berl, A. *et al.* (2010) Energy-efficient cloud computing. *Comput. J.*, **53**, 1045–1051.

[38] Agarwal, Y., Hodges, S., Chandra, R., Scott, J., Bahl, P. and Gupta, R. (2009) Somniloquy: Augmenting Network Interfaces to Reduce PC Energy Usage. *USENIX Symp. Networked Systems Design and Implementation*, Boston, USA, pp. 365–380. USENIX.

[39] Contreras, G. and Martonosi, M. (2005) Power Prediction for Intel XScale® Processors Using Performance Monitoring Unit Events. *Int. Symp. Low Power Electronics and Design (ISLPED'05)*, San Diego, USA, pp. 221–226. IEEE, Washington.