



Virt-B: Toward Performance Benchmarking of Virtual Machine Systems

Although cloud computing environments often use virtualization to implement resource sharing, the added hypervisor layer between virtual machines (VMs) and hardware resources creates performance overhead. Measuring virtualization performance is thus crucial for running applications efficiently and further improving VM techniques. To address the insufficiencies of existing benchmarking methods, the proposed three-layer strategy fully evaluates VM system performance and includes both a benchmark suite to measure various virtualization scenarios and an automated performance testing toolkit.

Kejiang Ye and Zhaohui Wu
Zhejiang University, China

Bing Bing Zhou
University of Sydney, Australia

Xiaohong Jiang
Zhejiang University, China

Chen Wang
Commonwealth Scientific and
Industrial Research Organisation,
Australia

Albert Y. Zomaya
University of Sydney, Australia

Widely used in today's Internet data centers and cloud computing platforms, virtualization improves resource utilization, reduces total ownership costs, and eases system resource management¹ by letting multiple virtual machines (VMs) share the resources of a single physical machine (PM). A VM monitor, or *hypervisor*, is a software layer between VMs and the underlying hardware that handles physical resource allocation for VMs. However, this indirect resource-access mechanism and the contention for resources among co-located VMs can both incur performance overhead. Quantifying this overhead is important for evaluating and comparing various virtualization technologies; a benchmarking tool is thus strongly needed to determine the costs of moving applications to a virtualized cloud computing environment.

Performance benchmarking of virtualized cloud computing environments entails several challenges. Traditional benchmarking methods don't account for virtualization characteristics, such as resource sharing and dynamic resource allocation, nor do they measure virtualization features such as live migration and virtual clusters. Moreover, existing benchmarks lack metrics for comprehensively understanding VM performance, especially fine-grained performance metrics such as the hypervisor's enter-and-exit operations and cache miss rates.

Here, we present a benchmarking tool aimed at addressing these challenges. Our three-layer performance benchmarking methodology can collect performance data from the hardware, hypervisor, and VM layers. We also introduce a new benchmark suite – Virt-B – that helps users understand the virtualization

Related Work in VM Performance Benchmarking

Many researchers have evaluated and modeled virtual machine (VM) performance. Timothy Wood and colleagues evaluated the basic virtualization overhead in terms of resource utilization and proposed a performance model to map the native system usage profile into a virtualized one.¹ Sajib Kundu and colleagues modeled the relationship between VM-host application performance and resources allocated to the VM to help users better size their VMs.² Our previous work performs a detailed performance comparison for different hypervisors with configurations varying from the single VM scenario (CPU, I/O, memory, and network) to a multiple VMs, or *virtual cluster*, scenario.³ Intel researchers focused on the server consolidation scenario, and found that resource interference could cause significant VM performance degradation.⁴ Similarly, several new benchmarks for virtualization environments — such as SPECvirt_sc2010 (www.spec.org/virt_sc2010/), VMmark (www.vmware.com/products/vmmark/), and vConsolidate⁵ — also focused only on the server consolidation scenario. Our benchmarking tool not only measures the performance loss in the server consolidation scenario, but also measures the performance loss under other new scenarios, including live migration and virtual clusters.

For performance analysis tools, researchers from HP proposed XenMon (<https://github.com/avsm/xen-unstable/tree/master/tools/xenmon>), a performance-profiling tool that investigates the relationship between an application's performance and its resource usage/requirements, especially for I/O-intensive applications. They also implemented Xenoprof (<http://xenoprof.sourceforge.net>), an extension of Oprofile in Linux that supports system profiling in the Xen virtual machine environment. Nezh Yigitbasi and colleagues implemented the C-Meter framework, which generates and submits workloads to the cloud to analyze cloud computing environments' performance.⁶ C-Meter is an extension of the GrenchMark grid workloads evaluation framework (<http://grenchmark.st.ewi.tudelft.nl>). None of these performance analysis tools can fully quantify the performance of VM systems, especially in relation to the hypervisor layer's performance overheads.

Most recently, Alexandru Iosup and colleagues proposed a generic approach for IaaS cloud benchmarking and discussed the challenges in developing such an approach.⁷ Carsten Binnig and colleagues pointed out that traditional benchmarks are insufficient for analyzing novel cloud services. They also presented some initial ideas for a new benchmark suitable for cloud computing.⁸ The Standard Performance Evaluation Corporation Research Group (<http://research.spec.org>) created a working group dedicated to developing a research benchmark for cloud computing. However, all of these efforts are still in the early stages.

Unlike existing work, our work focuses on the performance issues of VM systems and offers a scenario-driven approach for fully benchmarking their performance. Benchmarking VM system performance is critical to cloud computing, where VM techniques are commonly used.

References

1. T. Wood et al., "Profiling and Modeling Resource Usage of Virtualized Applications," *Proc. 9th ACM/IFIP/Usenix Int'l Conf. Middleware*, 2008, pp. 366–387.
2. S. Kundu et al., "Modeling Virtualized Applications Using Machine Learning Techniques," *Proc. 8th Int'l Conf. Virtual Execution Environments*, 2012, pp. 3–14.
3. K. Ye et al., "Performance Combinative Evaluation from Single Virtual Machine to Multiple Virtual Machine Systems," *Int'l J. Numerical Analysis & Modeling*, vol. 9, no. 2, 2012, pp. 351–370.
4. O. Tickoo et al., "Modeling Virtual Machine Performance: Challenges and Approaches," *ACM Sigmetrics Performance Evaluation Rev.*, vol. 37, no. 3, 2010, pp. 55–60.
5. P. Apparao et al., "Characterization and Analysis of a Server Consolidation Benchmark," *Proc. 4th ACM SIGPLAN/SIGOPS Int'l Conf. Virtual Execution Environments*, 2008, pp. 21–30.
6. N. Yigitbasi et al., "C-Meter: A Framework for Performance Analysis of Computing Clouds," *Proc. 9th IEEE/ACM Int'l Symp. Cluster Computing and the Grid*, 2009, pp. 472–477.
7. A. Iosup, R. Prodan, and D. Epema, "IaaS Cloud Benchmarking: Approaches, Challenges, and Experience," *Proc. Int'l Workshop on Hot Topics in Cloud Services*, 2012, pp. 1–2.
8. C. Binnig et al., "How is the Weather Tomorrow? Towards a Benchmark for the Cloud," *Proc. 2nd Int'l Workshop on Testing Database Systems*, 2009, article no. 9; doi: 10.1145/1594156.1594168.

overhead of typical scenarios, including single machine virtualization, server consolidation, VM mapping, live migration, and virtual clusters. Finally, we describe our toolkit implementation for automating the performance testing process and present a few case studies.

Requirements and Challenges

Performance analysis is the foundation of computer system design and optimization. With the wide adoption of virtualization technology in

cloud data centers, the demand is increasing for performance benchmarking in a virtualization environment. We show the motivation of VM performance benchmarking by analyzing the origins of virtualization overhead and summarizing the challenges of traditional benchmarks.

Origins of Overhead

The main costs associated with the hypervisor layer are its enter-and-exit operations, which can entail significant overhead, particularly for

I/O processing (including disk and network I/Os).² Different hypervisors implement I/O virtualization in different ways. Xen moves all the device drivers to Dom0 – an initial domain started by the Xen hypervisor on boot – and performs I/O processing on behalf of all guest VMs, while VMware installs device drivers in the hypervisor layer. Either approach results in some performance overhead.

Another cost source is resource contention among colocated VMs. Virtualization promises some isolation among colocated VMs, so that the execution of one VM doesn't affect another VM's performance. The isolation mechanism works well for CPU and memory sharing, but it's less effective for shared resources such as processor caches³ (some VMs might use a lot of cache space, which can eventually affect colocated VMs' performance).

The third cost source is complex virtualization operations such as live migration and virtual clusters, which incur performance overheads for the related VMs. Precopy migration – a technique that implements live migration of VMs and is widely used in both VMware⁴ and Xen hypervisor⁵ – creates several overheads:

- It consumes system resources, such as CPU in the source machine, to perform round-by-round page transmission.
- Data transmission between the source and destination machines consumes network bandwidth.
- Downtime in the migration process can cause temporary unavailability of service, which affects applications running in the migrating VM.

Other virtualization scenarios, such as cross-domain virtual clusters – that is, VMs in a virtual cluster running on more than one PM – can incur additional overheads for communication and data exchange among PMs as well.

Limitations of Traditional Benchmarking

Many traditional benchmarks exist for nonvirtualized environments. However, these benchmarking methods have certain limitations.

First, VMs and PMs behave differently when they run applications. In VMs, the hypervisor might interfere with application performance. Components such as front- and back-end drivers and the shadow page table introduce performance

complexity that traditional benchmarks can't fully reveal. To address this problem, we propose a three-layer benchmarking methodology to fully quantify performance characteristics from

- the hardware layer (such as cache misses);
- the hypervisor layer (such as hypervisor resource consumption and scheduling information); and
- the VM layers (such as workload performance changes).

Second, virtualization enables dynamic resource sharing. Traditional benchmarks often focus on static systems. We generalize six typical virtualization scenarios to measure the dynamic resource sharing performance of different virtualization technologies. We also design workloads and performance metrics for these scenarios. Our proposed scenario-based benchmark suite can thus give users insight into VM systems' performance issues.

Finally, the recently proposed benchmarks for virtualization environments such as SPECvirt_sc2010, VMmark, and vConsolidate have two limitations: they measure only the server consolidation scenario's performance and leave other important scenarios (such as live migration and virtual clusters) untouched; and they have fixed, inflexible benchmark workloads that can't be configured on demand to satisfy different user requirements.

A Three-Layer Benchmarking Methodology

Existing benchmarks usually measure system performance in a black-box way that provides only macro metrics, such as throughput and response time, and is insufficient for detecting underlying performance bottlenecks and their causes. Our three-layer performance benchmarking methodology comprehensively analyzes VM system performance.

Figure 1 shows the three layers: benchmarking, monitoring, and profiling. In the benchmarking layer, we evaluate the performance of typical virtualization scenarios by selecting, configuring, and rebuilding the existing benchmark workloads. From this layer, we collect macro performance metrics, such as application throughput and response time.

In the monitoring layer, we provide a fine-grained performance analysis using several existing

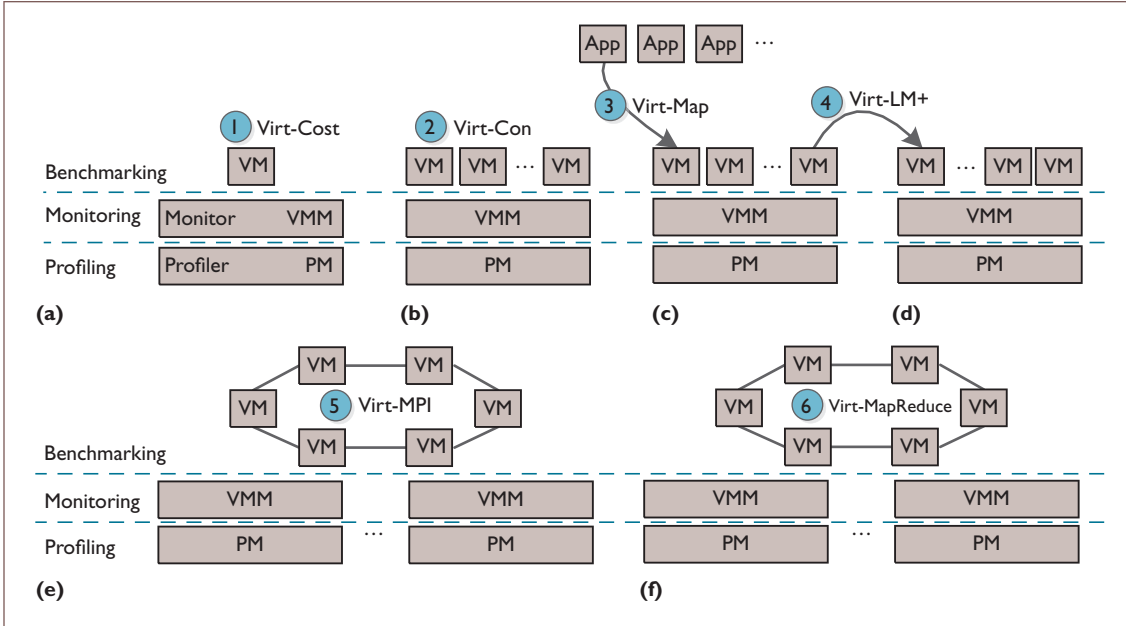


Figure 1. Typical virtualization scenarios and the three-layer benchmarking methodology. We defined six sub-benchmarks — *Virt-Cost*, *Virt-Con*, *Virt-Map*, *Virt-LM+*, *Virt-MPI*, and *Virt-MapReduce* — to measure virtualization performance in six primary application scenarios: (a) single-machine virtualization, (b) server consolidation, (c) virtual machine (VM) mapping, (d) VM live migration, (e) a computing-intensive virtual cluster, and (f) a data-intensive virtual cluster.

open source monitoring tools, including XenMon (<https://github.com/avsm/xen-unstable/tree/master/tools/xenmon>), Xentop ([http://wiki.xen.org/wiki/Xentop\(1\)](http://wiki.xen.org/wiki/Xentop(1))), and sysstat (<https://github.com/sysstat/sysstat>). This layer lets us easily obtain

- resource consumption information for each individual VM and hypervisor,
- the communication traffic and data exchanged between the VMs, and
- VM scheduling information across multiple hardware cores.

In the profiling layer, we provide finer-grained performance data such as hypervisor interference and shared resource contention. We profile hypervisor interference data by analyzing specific hypervisor operation events, such as hypervisor enter-and-exit operations. We profile shared resource competition data by periodically sampling various hardware events, such as CPU_CLK_UNHALTED, INST_RETIRED, LLC_MISSES, and DTLB_MISSES. We can then analyze the underlying overheads and discover potential performance bottlenecks.

All three layers collectively provide comprehensive performance data for an entire VM system.

Virt-B Benchmark Suite

Figure 1 shows virtualization technology’s six main application scenarios. We selected these scenarios because they are typical VM use cases in a cloud computing platform and reflect many mutually exclusive aspects of VM performance:

- *Single machine virtualization* quantifies the basic virtualization overhead.
- *Server consolidation* quantifies the consolidation’s isolation and interference.
- *VM mapping* quantifies the performance of different VM-to-PM mapping cases.
- *VM live migration* quantifies the live migration overheads of one or more VMs.
- *Virtual clusters* test both compute-intensive and data-intensive parallel workloads.

Our benchmark’s workloads and metrics are highly configurable and flexible; users can configure and generate new workloads with new metrics for particular new scenarios. As Table 1 shows, we also defined different sub-benchmarks for each scenario.

Virt-Cost measures the hypervisor’s basic virtualization overheads by comparing the performance with that of a native machine running the same configuration. To comprehensively

Table 1. Virt-B benchmark suite.

	Scenarios	Description	Workloads	Metrics
Virt-Cost	Single machine virtualization	Measures the basic virtualization overhead	Various workloads (CPU, memory, disk, network)	Performance slowdown (compared with native environment)
Virt-Con	Server consolidation	Measures the consolidation performance of VMs	Typical consolidation workloads (such as Java Server, file server, DB server, and Web server)	Performance isolation (PI) and consolidation capacity (CC)
Virt-Map	VM mapping	Measures the mapping performance of M -1 and M - N scenarios	Single machine workloads; parallel workloads	Overall throughput or response time of the applications deployed on M VMs
Virt-LM+	VM live migration	Measures the migration performance of VMs	Various workloads	Downtime, total migration time, amount of data transferred, workload performance slowdown
Virt-MPI	HPC virtual clusters	Measures the performance of HPC virtual clusters	Computing-intensive parallel workloads (such as HPCC and NPB)	Floating-point rate of execution, intra-cluster communication latency, and bandwidth
Virt-MapReduce	Hadoop virtual clusters	Measures the performance of Hadoop virtual clusters	Data-intensive parallel workloads (such as DFSIO, MRBench, and TeraSort)	Job completion time and distributed file system throughput

evaluate the virtualization efficiency, we select workloads with different characteristics, which include CPU-, memory-, disk I/O-, and network-intensive workloads. To measure the virtualization overheads, we use performance slowdown, defined as

$$PS = \frac{P_{pm} - P_{vm}}{P_{pm}},$$

where P_{pm} denotes the PM workload's baseline performance, and P_{vm} denotes the VM workload's performance. (Both machines have the exact same configuration.)

Virt-Con measures performance interference under different consolidation cases. Although several benchmarks exist for server consolidation, the workloads are fixed and can't be adjusted by users. *Virt-Con* offers a more flexible consolidation sub-benchmark that lets users configure specific workloads according to their requirements. To quantify the performance of server consolidation, we define two new metrics: performance isolation,

$$PI = \frac{1}{n} \sum_{i=1}^n \frac{W_i}{W'_i},$$

and consolidation capacity,

$$CC = \sum_{i=1}^n \frac{W_i}{W'_i},$$

where W_i denotes the performance of workload i running in consolidation, and W'_i denotes its performance running alone. PI is a metric to quantify the performance isolation between every two collocated VM workloads, while CC is a metric to quantify the consolidation capacity when more than two VM workloads are consolidated. Both metrics reflect server consolidation performance from different perspectives.

Virt-Map measures the performance of different VM-to-PM mapping cases. To evaluate mapping performance, we use two types of workloads. We use a single machine workload to evaluate the performance of mapping M VMs into 1 PM to observe the scalability. Here, M can scale from 1 to a threshold, such as 16. We use parallel workload to evaluate the performance of mapping M VMs into N PMs, which tells us how many PMs we need to support the M VMs. Here, N can scale from 1 to M . To compare the differences between different VM-to-PM

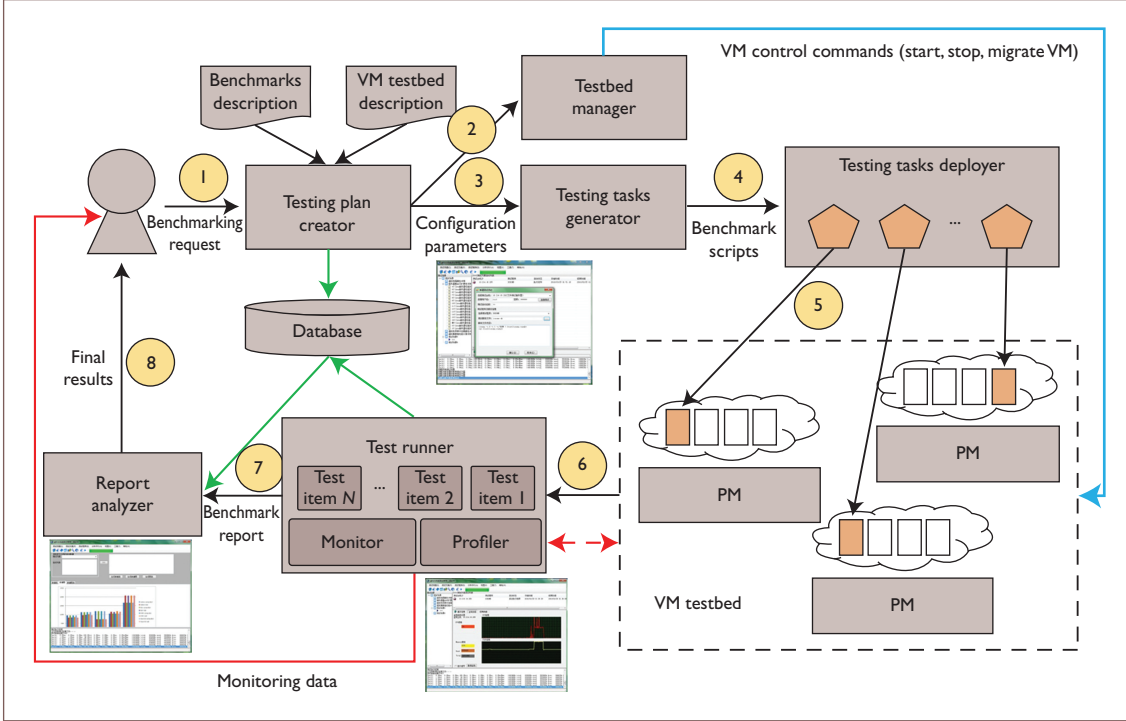


Figure 2. The automatic testing toolkit's workflow. The workflow consists of eight steps, from the testing plan creator receiving a user request (step 1) to the report analyzer visualizing the report's data and comparing results (step 8).

mapping cases, we use overall throughput or the response time of the applications deployed on M VMs.

Virt-LM+ measures the live migration performance of VMs. We extend our original migration benchmark *Virt-LM*⁶ to *Virt-LM+* to quantify the migration performance of both a single VM and multiple VMs (including virtual clusters) with four metrics: downtime, total migration time, the amount of data transferred, and workload performance slowdown. Here, we define the workload performance slowdown in migration as

$$MPS = \frac{P_{norm} - P_{mig}}{P_{norm}},$$

where P_{norm} denotes the baseline performance of a normal workload (without migration) and P_{mig} denotes the performance of a workload running in a migrating VM.

Virt-MPI measures the performance of a computing-intensive virtual cluster with workloads such as HPC Challenge Benchmark (HPCC) or NAS Parallel Benchmark (NPB) running in a message-passing interface (MPI) parallel environment. *Virt-MPI* measures performance

using floating-point rate of execution, intra-cluster communication latency, and bandwidth.

Virt-MapReduce measures the performance of a data-intensive application running in a virtual cluster, typically with MapReduce workloads, such as MRBench, TeraSort, and DFSIO. Here, the measures are a MapReduce job's completion time and the underlying distributed file system's throughput.

To comprehensively quantify virtualization performance, you must run all six sub-benchmarks; to investigate a specific scenario's performance, you can run the sub-benchmarks individually. As with existing benchmarks such as HPCC, we show users the scores for each sub-benchmark.

Automatic Testing Toolkit

The testing process can be complex and time-consuming because many scenarios are involved in some testing cases. To facilitate testing and reduce manual intervention, we designed and implemented an automatic performance testing toolkit (available at <http://dartcloud.zju.edu.cn/benchmarking/toolkit.html>). Figure 2 shows the system modules and workflow.

The toolkit's main features include support for

- testbed management, including configuring, creating, migrating, and shutting down VMs;
- workload management, including selecting, configuring, and rebuilding workloads for various scenarios; and
- testing-process management, including deploying and running the workloads, and fetching and analyzing the reports in an automatic way.

In keeping with our three-layer benchmark methodology, we also implement Monitor and Profiler modules for the monitoring and profiling layers, respectively, to provide insight into the internal VM system.

As Figure 2 shows, the benchmarking workflow consists of eight steps:

1. A user sends a benchmarking request to the *testing plan creator* to create a testing plan, which can be stored in the database and reused in the future.
2. The testing plan creator sends the testbed requirement – such as 16 VMs, each of which has a virtual CPU (VCPU) and 1 Gbyte DRAM – to the *testbed manager*, which then sets up and configures a specific VM testbed by remotely managing the virtualized resource pool.
3. The testing plan creator sends the workload requirement to the *testing tasks generator*, with specified workload types and their parameters.
4. The testing tasks generator creates detailed testing tasks using python scripts that meet the configuration parameters (such as workload types/parameters).
5. The *testing tasks deployer* sends the task scripts to dedicated VMs.
6. The *test runner* controls the remote scripts' execution. If a testing script has sequential items, they'll be run one by one. At the same time, the *monitor* measures the testbed's resource utilization and updates the user interface every five seconds (or other user-specified time intervals). The *profiler* collects the profiling data using event counters. Profiles are used for offline performance analysis and performance prediction.
7. When all test items are complete, benchmark reports are sent to the *report analyzer* for further analysis and stored simultaneously in the database.
8. The report analyzer can visualize the report's data and compare results. Users can also retrieve the original reports for manual analysis.

Although the current implementation of our toolkit supports the Xen VM system, it can be easily extended for benchmarking the performance of other VM systems, such as Kernel-based Virtual Machine (KVM) and OpenVZ.

Case Studies

We conducted three case studies of our benchmarking tool on server consolidation, live migration, and virtual clusters, respectively. We performed all experiments on Dell T710 servers, with two Quad-core 64-bit Xeon processors E5620 at 2.4 GHz, 32 Gbyte DRAM, and 1 Gbyte Ethernet. We created the virtualization environment based on Xen 3.3.1 and configured each VM with 1 VCPU and 1 Gbyte DRAM.

Server Consolidation

Figure 3 shows the consolidation performance of four typical server workloads for an Internet data center. The Virt-Con sub-benchmark and the toolkit can easily consolidate any two workloads; this isn't possible with existing consolidation benchmarks because they can't dynamically mix workloads.

In our experiment, we could measure the performance interference and consolidation capacity of any two workloads. As the figure shows, colocating one VM running SPECjbb and one running Sysbench leads to the best performance isolation and achieves the highest consolidation capacity. Because SPECjbb is CPU-intensive and Sysbench is memory-intensive, their complementary resource use leads to good performance.

With Virt-Con, users can study more complex workload-consolidation characteristics and select the best strategies to suit their workloads.

Live Migration

Live migration lets VMs move from one PM to another; it's a key technology for cloud computing platforms because it lets them dynamically manage their resources. Understanding live migration performance is important for

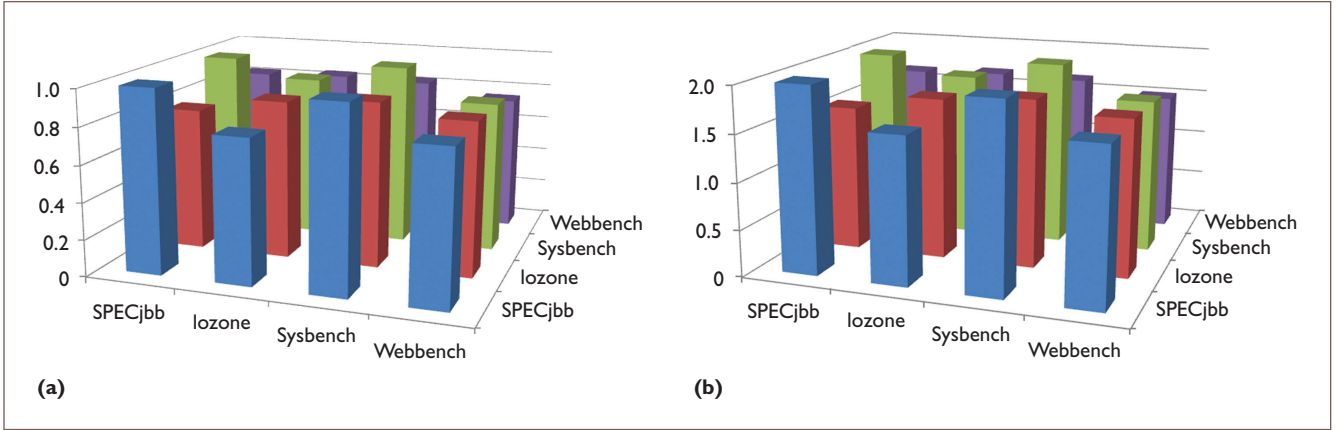


Figure 3. Server consolidation performance for four of an Internet data center's typical server workloads. Colocating one VM running SPECjbb and one running Sysbench leads to (a) the best performance isolation and (b) achieves the highest consolidation capacity.

Table 2. Live migration performance.

Workloads	Downtime (ms)	Migration time (sec)	Data transferred (pages)	Performance loss (%)
SPECjbb	205	31.39	867,698	18.91
IOzone	245	33.47	727,082	53.68
Sysbench	35	10.53	271,322	34.52
Webbench	36	10.65	274,683	24.57

Table 3. Virtual cluster performance.

	1 VM	5 VMs	10 VMs	15 VMs
HPC virtual cluster's network latency (μ s)	20.37	36.67	39.78	40.22
Hadoop virtual cluster's execution time (sec)	490	623	666	677

developing better migration and resource-management algorithms.

Table 2 compares Virt-LM+'s live migration and nonmigration performances on four measures: downtime, migration time, amount of data transferred, and workload performance loss. SPECjbb and IOzone have relatively poor migration performance, with longer downtimes and more data transfers during migration. This is because SPECjbb and IOzone trigger many memory swap operations and generate many dirty pages, which affects the downtime and migration time.

Virtual Clusters

Virtual clusters are another popular virtualization scenario that let users quickly establish a cluster environment for specific use-cases, such as master-worker-based parallel computing. We studied the performance of two kinds of virtual clusters:

- a compute-intensive virtual cluster, running an HPCC benchmark workload in an MPICH2 parallel environment; and
- a data-intensive virtual cluster, running a wordcount benchmark in a MapReduce parallel environment.

Table 3 shows the partial results produced by the Virt-MPI and Virt-MapReduce sub-benchmarks. The HPC VC metric is the cluster network latency, whereas the Hadoop VC metric is the running wordcount's execution time on a 500 Mbyte text file. When we increase the virtual cluster size, we see an increase in both the HPC virtual cluster's network latency and the Hadoop virtual cluster's execution time. This is because, when we add more nodes, the network virtualization overheads increase despite the execution time decrease expected from parallelism.

Our scenario-based three-layer benchmark method is an effective and efficient solution to comprehensively quantify the performance overheads and detect potential performance bottlenecks of VM systems. However, it's also a general and flexible solution. We can easily extend this benchmark method to other performance-critical systems, such as emerging big data systems. Our future work will include extending our method to more complex systems and making the performance benchmarking a service in the cloud, accessible to users across the Internet. ☐

Acknowledgment

The National High Technology Research 863 Major Program of China (no. 2011AA01A207) and the National Natural Science Foundation of China (no. 61272128) support this work.

References

1. B. Sotomayor et al., "Virtual Infrastructure Management in Private and Hybrid Clouds," *IEEE Internet Computing*, vol. 13, no. 5, 2009, pp.14–22.
2. G. Dasgupta et al., "Workload Management for Power Efficiency in Virtualized Data Centers," *Comm. ACM*, vol. 54, no. 7, 2011, pp.131–141.
3. Y. Koh et al., "An Analysis of Performance Interference Effects in Virtual Environments," *IEEE Int'l Symp. Performance Analysis of Systems and Software*, 2007, pp. 200–209.
4. M. Nelson et al., "Fast Transparent Migration for Virtual Machines," *Proc. Usenix Ann. Technical Conf.*, 2005, p. 25.
5. C. Clark et al., "Live Migration of Virtual Machines," *Proc. 2nd Symp. Networked Systems Design and Implementation*, 2005, pp. 273–286.
6. D. Huang et al., "Virt-LM: A Benchmark for Live Migration of Virtual Machine," *Proc. ACM/SPEC Int'l Conf. Performance Eng.*, 2011, pp. 307–316.

Kejiang Ye is a PhD candidate at Zhejiang University and a visiting PhD student at the University of Sydney, Australia. His research interests include virtualization and cloud computing, performance/energy analysis, modeling, and optimization. Ye has a BSc in software engineering from Zhejiang University, China. Contact him at yekejiang@zju.edu.cn.

Zhaohui Wu is a professor in the Department of Computer Science at Zhejiang University. His research interests include grid computing, service computing, distributed artificial intelligence, and pervasive computing. Wu has


a PhD in computer science from Zhejiang University, China. He is a Standing Council Member of the China Computer Federation and a senior member of IEEE. Contact him at wzhu@zju.edu.cn (corresponding author).

Bing Bing Zhou is an associate professor in the School of Information Technologies at the University of Sydney. His research interests include parallel/distributed computing, grid and cloud computing, peer-to-peer systems, parallel algorithms, and bioinformatics. Zhou has a PhD in computer science from Australian National University. Contact him at bing.zhou@sydney.edu.au.

Xiaohong Jiang is an associate professor in the Department of Computer Science at Zhejiang University. Her research interests include distributed systems, virtual environment, and data service. Jiang has a PhD in computer science from Zhejiang University. Contact her at jiangxh@zju.edu.cn.

Chen Wang is a senior research scientist at the Commonwealth Scientific and Industrial Research Organisation (CSIRO) Computational Informatics in Australia. His research interests include distributed, parallel, and trustworthy systems, with a focus on resource management in cloud computing, accountable distributed systems, and demand response algorithms in the smart grid. Wang has a PhD in computer science from Nanjing University. Contact him at chen.wang@csiro.au.

Albert Y. Zomaya is the Chair Professor of High Performance Computing & Networking in the School of Information Technologies at the University of Sydney. His research interests include parallel and distributed systems and green computing. Zomaya has a PhD in control engineering from Sheffield University, England. He is editor in chief of *IEEE Transactions on Computers*, an associate editor for 20 leading journals, and a fellow of the American Association for the Advancement of Science, IEEE, and the Institution of Engineering and Technology UK. Contact him at albert.zomaya@sydney.edu.au.

 Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.