Quantitative Comparison of Xen and KVM

Todd Deshane, Zachary Shepherd, Jeanna N. Matthews
Computer Science
Clarkson University
Potsdam, NY 13699 USA
{deshantm, shephezj, jnm}@clarkson.edu

Muli Ben-Yehuda
IBM Haifa Research Lab
Haifa University Campus
Mount Carmel, Haifa 31905
Israel
muli@il.ibm.com

Amit Shah
Qumranet
B-15, Ambience Empyrean
64/14 Empress County
Behind Empress Gardens
Pune 411001 India
amit.shah@ qumranet.com

Balaji Rao
National Institute of Technology Karnataka
Surathkal 575014 India
balajirrao@gmail.com

ABSTRACT
We present initial results from and quantitative analysis of two leading open source hypervisors, Xen and KVM. This study focuses on the overall performance, performance isolation, and scalability of virtual machines running on these hypervisors. Our comparison was carried out using a benchmark suite that we developed to make the results easily repeatable. Our goals are to understand how the different architectural decisions taken by different hypervisor developers affect the resulting hypervisors, to help hypervisor developers realize areas of improvement for their hypervisors, and to help users make informed decisions about their choice of hypervisor.

1. INTRODUCTION
IT professionals, developers, and other users of virtualization on Linux often look for quantitative results to compare their hypervisor options. In this study, we compare two open source hypervisors: the established Xen hypervisor and the more recent Kernel-based Virtual Machine (KVM) driver.

Since its public release in 2003, Xen has been the subject of many performance comparisons [3, 4, 6, 14, 15, 28, 29, 30, 35, 46, 48, 49]. Xen is well-known for its near-native performance and its use of paravirtualization. KVM, a relative new-comer to the virtualization market with its debut in early 2007, relies on CPU support for virtualization and leverages existing Linux kernel infrastructure to provide an integrated hypervisor approach (as opposed to Xen's stand-alone hypervisor approach). KVM is known for its rapid inclusion into the mainline Linux kernel. As KVM matures, more performance testing and comparisons are being done with it, like those at IBM [21].

With the wide variety of virtualization options available, several efforts to provide benchmarks specifically designed for comparing different virtualization systems have been initiated [8, 12, 27, 40, 42]. For this study, we developed an open source virtualization benchmark suite named benchvm [8] to help automate testing, including setting up the guests and running some of the tests. Our goal in using and developing benchvm has been to provide repeatability and transparency so that others can easily validate the results. The benchvm suite is still under heavy development and, although still useful, should not yet be considered production-ready.

For our initial set of tests, the experimental setup consisted of Ubuntu Linux 8.04 AMD64 on the base machine. The Linux kernel 2.6.24-18, Xen 3.2.1+2.6.24-18-xen, and KVM 62 were all installed from Ubuntu packages. All guests were automatically created by a benchvm script that called debootstrap and installed Ubuntu 8.04 AMD64. The guests were then started with another benchvm script that passed the appropriate kernel (2.6.24-18-xen for Xen and 2.6.24-18 for KVM). The hardware system was a Dell OptiPlex 745 with a 2.4 GHz Intel Core 2 CPU 6600, 4 GB of RAM, 250 GB hard drive, and two 1 Gigabit Ethernet cards. Test results from more software versions and hardware configurations are reported on our Benchvm Results website [9].

Our benchmark testing focuses on three pillars of virtualization benchmarking: overall performance, performance isolation, and scalability. We discuss the testing process and present our quantitative results from the tests in each of these categories. Due to space limitations, we then briefly mention related work and list it as further reading at the end of this paper.

2. OVERALL PERFORMANCE
To measure overall system performance, we ran a CPU-intensive test, a kernel compile, an IOzone [22] write test, and an IOzone read test. We compared the Xen and KVM numbers against the non-virtualized (native) Linux baseline, shown in Table 1.

For the CPU-intensive test, Xen was very close to Linux and KVM had slightly more degradation than Xen. For the kernel compile, the degradation for Xen was about half that of Linux (likely due to less memory). KVM again had slightly more degradation than Xen. On the other hand, KVM had higher write and read performance than Xen according to our results. We believe that KVM may have performed better than Xen in terms of I/O due to disk caching.

The Phoronix Test Suite [37] was useful for running and publishing the kernel compile and IOzone tests. Additional performance results including running the Phoronix Test Suite's Universe Command Line Interface (CLI) tests with the command line parameter `universe-cli`, and testing on other platforms and with other benchmarks including Bonnie++ [11], Iperf [23], and Netperf [34], are reported on the Benchvm Results website [9].

Xen Summit, June 23-24, 2008, Boston, MA, USA.

Table 1. Overall performance of base Linux, Xen, and KVM

	Linux	Xen	KVM
CPU	1.000	0.999	0.993
Kernel Compile	1.000	0.487	0.384
IOzone Write	1.000	0.855	0.934
IOzone Read	1.000	0.852	0.994

3. PERFORMANCE ISOLATION

Performance isolation is a measure of how well guests are protected from extreme resource consumption in other guests. We used the testing methodology and isolation benchmark suite that some of the authors of this paper developed previously [28]. For the isolation tests in this study, we ran SPECweb2005 [43] on four virtual machine clients. The guest that runs a stress test is referred to as the Stressed VM, since it is under a significant load specific to the type of resource being tested. We measured the percent of degradation in good response rate for the SPECweb clients running the support workload with the stress test versus the baseline without the stress test.

In Table 2, we show the results of the performance isolation tests for Xen and KVM. Degradation of the Stressed VM is expected. Isolation problems are indicated by degradation in the performance of the Normal VM. Low degradation percentages are better and DNR is the worst possible percent degradation. DNR means that the guest "did not return" results and usually indicates a kernel panic or network problem for the guest.

Xen shows good isolation properties for the memory, fork, CPU, and disk stress tests as seen in the Normal VM column. Xen shows very little isolation for the network sender and no isolation for the network receiver. Xen shows unexpectedly good performance for the disk test and unexpectedly poor performance for the network sender test.

KVM shows good isolation properties for all of the stress tests and unexpectedly good performance for the network sender. However, KVM shows unexpectedly poor performance for the disk test and the network receiver test.

Table 2. Performance isolation of Xen versus KVM
Higher degradation percentages are bad and DNR is the worst possible.

	Xen		KVM	
	Stressed VM	Normal VM	Stressed VM	Normal VM
Memory	DNR	0	DNR	0
Fork	DNR	0	DNR	0
CPU	0	0	0	0
Disk	3	0.03	22.2	0.01
Network receiver	0.27	0.7	DNR	0.12
Network sender	2.53	2.08	0	0

4. SCALABILITY

A virtualization system's level of scalability is determined by its ability to run more virtual machines without loss of performance. To measure scalability in this study, we first compiled Apache source code on one guest and then we increased the number of guests that were each doing an Apache compile. In the following graphs, lower compile times (except 0) and more simultaneous guests indicate better scalability. Gaps in the graph (compile times of 0 seconds) indicate that the guests crashed and therefore were unable to report results.

For Xen, in Figure 1, as we increased the number of guests, the time to compile Apache increased at a linear rate compared to the number of guests. This shows that Xen had excellent scalability and that Xen was able to share resources among guests well.

For KVM, in Figure 2, as we increased the number of guests to 4, 1 of the four guests crashed. As the guests were increased to 8, 4 guests crashed. With 16 guests, 7 guests crashed. With 30 guests, the system crashed during the compile. This indicates that KVM was not able to maintain performance as the number of guests increased.

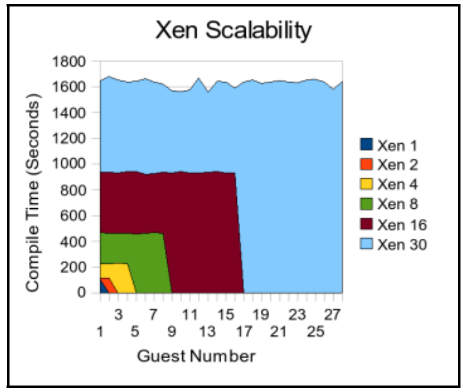

Figure 1. Scalability of building Apache on Xen guests
Higher compile times are bad and more simultaneous guests are better.

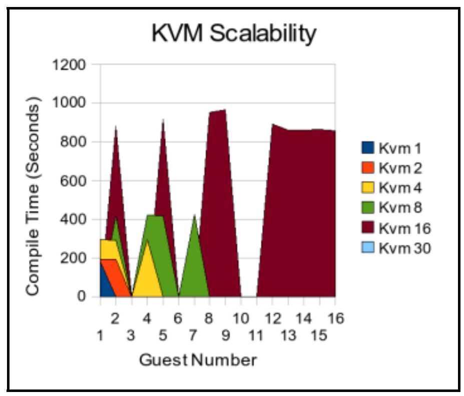

Figure 2. Scalability of building Apache on KVM guests
Higher compile times are bad and more simultaneous guests are better.
A compile time of 0 seconds indicates that the guest crashed (did not report results).

5. RELATED WORK

There are countless performance studies on virtualization, including [2, 4, 6, 10, 14, 15, 17, 19, 20, 21, 25, 26, 28, 29, 30, 31, 33, 35, 36, 39, 41, 44, 46, 47, 48, 49]. In addition to our benchvm test suite [8], other virtualization benchmark suites include vConsolidate [3, 12], VMmark [27], and Virtbench [40]. There are a number of general test suites, test harnesses, and related tools such as the Autotest Framework [5], BCFG2 [7], CFengine [13], DejaGnu [16], Expect [18], Kvm-test [24], Phoronix Test Suite [37], Puppet [38], Tcltest [45], and Xm-test [50]. General performance studies are in [1, 32].

6. CONCLUSION

We have presented a quantitative comparison of Xen and KVM focusing on overall performance, performance isolation, and scalability. The most striking difference between the two systems was in scalability. KVM had substantial problems with guests crashing, beginning with 4 guests. KVM had better performance isolation than Xen, but Xen's isolation properties were also quite good. The overall performance results were mixed, with Xen outperforming KVM on a kernel compile test and KVM outperforming Xen on I/O-intensive tests. We would like to extend our comparison to include Xen with full virtualization (HVM) and KVM with paravirtualized I/O.

7. ACKNOWLEDGMENTS

We acknowledge Wenjin Hu and Madhujith Hapuarachchi for their Master's thesis work on performance isolation and scalability benchmarking. We would also like to thank Cyrus Katrak and Martin McDermott for early development and testing of benchvm. Lastly, we very much appreciate the feedback and support of the Xen and KVM communities.

8. REFERENCES

[1] K. Adams and O. Agesen, *A Comparison of Software and Hardware Techniques for x86 Virtualization*, International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2006.

[2] P. Apparao, R. Iyer, and D. Newell, *Towards Modeling & Analysis of Consolidated CMP Servers*, Workshop on the Design, Analysis, and Simulation of Chip Multi-Processors (dasCMP), 2007.

[3] P. Apparao, R. Iyer, X. Zhang, D. Newell, and T. Adelmeyer, *Characterization & Analysis of a Server Consolidation Benchmark*, ACM/USENIX International Conference on Virtual Execution Environments (VEE), 2008.

[4] P. Apparao, S. Makineni, and D. Newell, *Characterization of Network Processing Overheads in Xen*, IEEE International Workshop on Virtualization Technology in Distributed Computing (VTDC), 2006.

[5] Autotest Framework, http://test.kernel.org/autotest.

[6] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, *Xen and the Art of Virtualization*, ACM Symposium on Operating Systems Principles (SOSP), 2003.

[7] BCFG2, http://www.bcfg2.org/.

[8] Benchvm, http://code.google.com/p/benchvm/.

[9] Benchvm Results, http://www.clarkson.edu/projects/virtualization/benchvm/.

[10] H.K.F. Bjerke, R.J. Andresen, and J. Amundsen, *Virtualization in Clusters*, Course Project, Norwegian University of Science and Technology (NTNU), http://haavard.dyndns.org/virtualization/clust_virt.pdf, 2004.

[11] Bonnie++, http://www.coker.com.au/bonnie++/.

[12] J.P. Casazza, M. Greenfield, and K. Shi, *Redefining Server Performance Characterization for Virtualization Benchmarking*, Intel Technology Journal, 2006.

[13] CFengine, http://www.cfengine.org/.

[14] L. Cherkasova and R. Gardner, *Measuring CPU Overhead for I/O Processing in the Xen Virtual Machine Monitor*, USENIX Annual Technical Conference, 2005.

[15] B. Clark, T. Deshane, E. Dow, S. Evanchik, M. Finlayson, J. Herne, and J.N. Matthews, *Xen and the Art of Repeated Research*, USENIX Annual Technical Conference, 2004.

[16] DejaGnu, http://www.gnu.org/software/dejagnu/.

[17] U. Drepper, *The Cost of Virtualization*, ACM Queue Magazine, 2008.

[18] Expect, http://expect.nist.gov/.

[19] S.S. Foley, V. Pandey, M. Tang, F. Terkhorn, and A. Venkatraman, *Benchmarking Servers using Virtual Machines*, Indiana University, http://www.cs.indiana.edu/~mhtang/paper.pdf, 2007.

[20] D. Gupta, R. Gardner, and L. Cherkasova, *XenMon: QoS Monitoring and Performance Profiling Tool*, Technical Report, Internet Systems and Storage Laboratory at HP Laboratories, 2005.

[21] R. Harper and K. Rister, *KVM Limits: Arbitrary or Architectural?*, Presentation, KVM Forum, http://kvm.qumranet.com/kvmwiki/KvmForum2008?action=AttachFile&do=get&target=kdf2008_6.pdf, 2008.

[22] IOzone Filesystem Benchmark, http://www.iozone.org/.

[23] Iperf, http://iperf.sourceforge.net/.

[24] Kvm-test, http://kvm.qumranet.com/kvmwiki/KVMTest.

[25] J. LeVasseur, V. Uhlig, M. Chapman, P. Chubb, B. Leslie, and G. Heiser, *Pre-virtualization: Soft Layering for Virtual Machines*, Technical Report, Fakultat fur Informatik, Universitat Karlsruhe (TH), 2006.

[26] J. Liu, W. Huang, B. Abali, and D.K. Panda, *High Performance VMM-Bypass I/O in Virtual Machines*, USENIX Annual Technical Conference, 2006.

[27] V. Makhija, B. Herndon, P. Smith, L. Roderick, E. Zamost, and J. Anderson, *VMmark: A Scalable Benchmark for Virtualized Systems*, Technical Report, VMware, 2006.

[28] J.N. Matthews, W. Hu, M. Hapuarachchi, T. Deshane, D. Dimatos, G. Hamilton, M. McCabe, and J. Owens, *Quantifying the Performance Isolation Properties of Virtualization Systems*, ACM Workshop on Experimental Computer Science (ExpCS), http://www.clarkson.edu/class/cs644/isolation/, 2007.

[29] A. Menon, A.L. Cox, and W. Zwaenepoel, *Optimizing Network Virtualization in Xen*, USENIX Annual Technical Conference, 2006.

[30] A. Menon, J.R. Santos, Y. Turner, G.J. Janakiraman, and W. Zwaenepoel, *Diagnosing Performance Overheads in the Xen Virtual Machine Environment*, ACM/USENIX International Conference on Virtual Execution Environments (VEE), 2005.

[31] Microsoft, *Comparing Web Service Performance: WS Test 1.5 Benchmark Results for .NET 3.5/Windows Server 2008 vs. IBM WebSphere 6.1/Red Hat Linux Advanced Platform 5*, http://msdn.microsoft.com/en-us/netframework/cc302396.aspx, 2008.

[32] J.C. Mogul, *Brittle Metrics in Operating Systems Research*, IEEE Workshop on Hot Topics in Operating Systems (HotOS), 1999.

[33] K.T. Moller, *Virtual Machine Benchmarking*, Diploma Thesis, Universitat Karlsruhe (TH), 2007.

[34] Netperf, http://www.netperf.org/netperf/.

[35] D. Ongaro, A.L. Cox, and S. Rixner, *Scheduling I/O in Virtual Machine Monitors*, ACM/USENIX International Conference on Virtual Execution Environments (VEE), 2008.

[36] P. Padala, K.G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, *Adaptive Control of Virtualized Resources in Utility Computing Environments*, ACM European Conference on Computer Systems (EuroSys), 2007.

[37] Phoronix Test Suite, http://www.phoronix-test-suite.com/.

[38] Puppet, http://www.reductivelabs.com/projects/puppet/.

[39] B. Quetier, V. Neri, and F. Cappello, *Selecting A Virtualization System For Grid/P2P Large Scale Emulation*, Workshop on Experimental Grid Testbeds for the Assessment of Large-Scale Distributed Applications and Tools (EXPGRID), 2006.

[40] R. Russell, Virtbench, http://www.ozlabs.org/~rusty/virtbench/.

[41] S. Soltesz, H. Potzl, M.E. Fiuczynski, A. Bavier, and L. Peterson, *Container-based Operating System Virtualization: A Scalable, High-performance Alternative to Hypervisors*, ACM European Conference on Computer Systems (EuroSys), 2007.

[42] *SPEC to Develop Standard Methods of Comparing Virtualization Performance*, Press Release, http://www.spec.org/specvirtualization/pressrelease.html, 2006.

[43] SPECweb2005, http://www.spec.org/web2005/.

[44] Sun Microsystems, *Web Services Performance Comparing Java 2 Enterprise Edition (J2EE platform) and .NET Framework*, http://java.sun.com/performance/reference/whitepapers/WS_Test-1_0.pdf, 2004.

[45] Tcltest, http://www.tcl.tk/man/tcl/TclCmd/tcltest.htm.

[46] A. Theurer, K. Rister, O. Krieger, R. Harper, and S. Dobbelstein, *Virtual Scalability: Charting the Performance of Linux in a Virtual World*, Ottawa Linux Symposium, Volume 2, http://www.linuxsymposium.org/2006/proceedings.php, 2006.

[47] A. Venkatraman, V. Pandey, B. Plale, and S.S. Shei, *Benchmarking Effort of Virtual Machines on Multicore Machines*, Technical Report, Indiana University, http://www.cs.indiana.edu/Research/techreports/, 2007.

[48] VMware, *A Performance Comparison of Hypervisors*, http://www.vmware.com/pdf/hypervisor_performance.pdf, 2007.

[49] XenSource, *A Performance Comparison of Commercial Hypervisors*, http://www.xensource.com/Documents/hypervisor_performance_comparison_1_0_5_with_esx-data.pdf, 2007.

[50] Xm-test, http://www.xen.org/files/summit_3/xs0906-xmtest.pdf.