USENIX Association

Proceedings of the FREENIX Track: 2004 USENIX Annual Technical Conference

Boston, MA, USA
June 27–July 2, 2004

Xen and the Art of Repeated Research

Bryan Clark, Todd Deshane, Eli Dow, Stephen Evanchik, Matthew Finlayson, Jason Herne, Jeanna Neefe Matthews
Clarkson University
{clarkbw, deshantm, dowem, evanchsa, finlayms, hernejj, jnm}@clarkson.edu

Abstract

Xen is an x86 virtual machine monitor produced by the University of Cambridge Computer Laboratory and released under the GNU General Public License. Performance results comparing XenoLinux (Linux running in a Xen virtual machine) to native Linux as well as to other virtualization tools such as User Mode Linux (UML) were recently published in the paper "Xen and the Art of Virtualization" at the Symposium on Operating Systems Principles (October 2003). In this study, we repeat this performance analysis of Xen. We also extend the analysis in several ways, including comparing XenoLinux on x86 to an IBM zServer. We use this study as an example of repeated research. We argue that this model of research, which is enabled by open source software, is an important step in transferring the results of computer science research into production environments.

1. Introduction

Repeated research is a well-respected model of investigation in many sciences. Independent tests of published research are valued because they document the general applicability of results. In addition, repeated research often sheds new light on aspects of a work not fully explored in the original publication.

In computer science, however, it is most common for researchers to report results from testing the software that they themselves have implemented. There are many reasons for this, including the wide variety of hardware and software platforms and the difficulty transferring fragile research software to a new environment. However, without independent trials, it is difficult to establish reported experience as repeatable fact.

Computer systems researchers often note with dismay the number of great ideas that are not incorporated into production computer systems. We argue that encouraging repeated research is an important step towards this transfer of technology. Researchers who release their code to the open source community make a valuable step towards encouraging repeated research in computer science.

In this paper, we present results that repeat and extend experiments described in the paper "Xen and Art of Virtualization" by Barham et al. from SOSP-03. [Xen03]. Xen is an x86 virtual machine monitor produced by the University of Cambridge Computer Laboratory in conjunction with Microsoft Research and Intel Research. Xen has been released under the GNU General Public License at xen.sourceforge.net.

In [Xen03], Barham et al. explore the performance of XenoLinux – Linux running in Xen. They compare performance to native Linux as well as to other virtualization tools such as User Mode Linux (UML) and VMWare Workstation. They also examine how the performance of Xen scales as additional guest operating systems are created.

In this paper, we first report the results of repeating measurements of native Linux, Xenolinux and User Mode Linux on hardware almost identical to that used in the Xen paper. Second, we present results comparing Xen to native Linux on a less powerful PC. Third, we evaluate Xen as a platform for virtual web hosting. Fourth, we compare the performance of benchmarks running in XenoLinux to the same benchmarks running in Linux on an IBM zServer that we won as a prize in the 2001 IBM Linux Scholar Challenge competition. Finally, we discuss our general experiences with repeated research.

We structure the rest of our paper around a set of questions and their answers.

- Can we reproduce the results from the SOSP-03 Xen paper?
- Could we realistically use Xen for virtual web hosting?
- Do you need a $2500 Dell Xeon Server to run Xen effectively, or will a 3 year old x86 do the job?

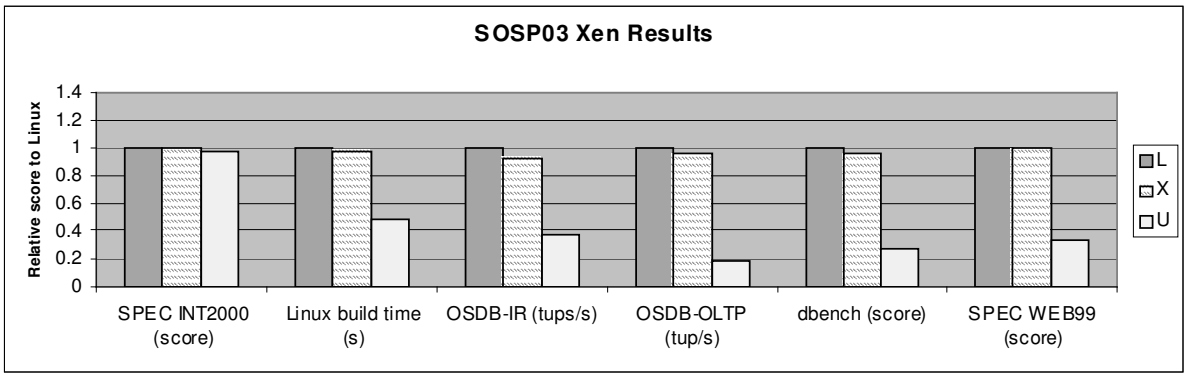
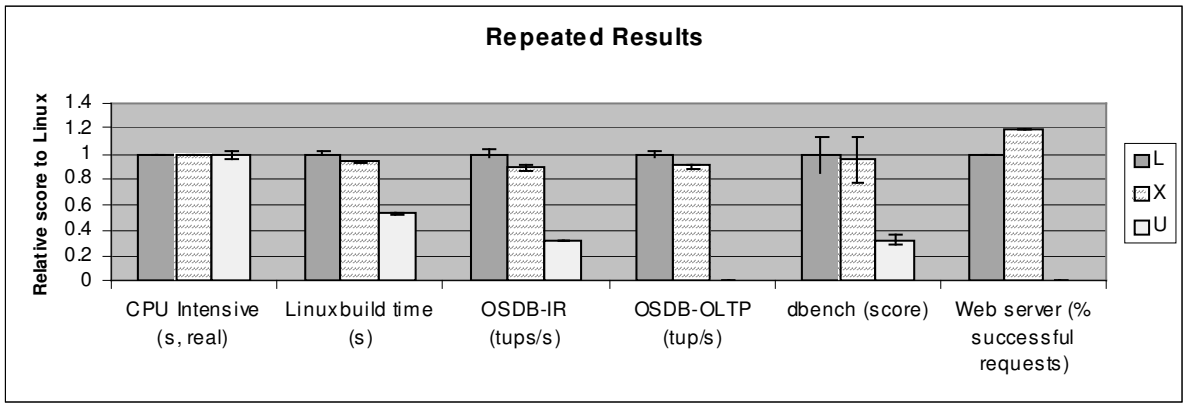


Figure 1a Relative Performance of Native Linux (L), XenoLinux (X) and User-Mode Linux (U). This data is from Figure 3 of [Xen03].



- How does a virtual machine monitor on commodity PCs compare to a virtual machine monitor on a mainframe specifically designed to support virtualization?
- What have we learned about repeated research?

2. Repeat the SOSP-03 Performance Analysis of Xen

Our first task was to convince ourselves that we could successfully reproduce the results presented in [Xen03]. The paper itself contained clear details on their test machine – a Dell 2650 dual processor 2.4GHz Xeon server with 2 GB RAM, a Broadcom Tigon 3 Gigabit Ethernet NIC and a single Hitachi DK32EJ 146 GB 10K RPM SCSI disk.

We had little trouble acquiring a matching system. We ordered a machine matching their specifications from Dell for approximately $2000. If we had been repeating older research, reproducing an acceptable hardware platform might have been a significant challenge.

The only significant difference in our system was the SCSI controller. Their controller had been a 160 MB/sec DELL PERC RAID 3Di and ours was a 320 MB/sec Adaptec 29320 aic79xx. Thus our first hurdle was the need to port the driver for our SCSI controller to Xen. The Xen team was extremely helpful in this process and in the end we contributed this driver (and several others) back into the Xen source base.

Our second hurdle was assembling and running all of the benchmarks used in the Xen paper including OSDB, dbench, lmbench, ttcp, SPEC INT CPU 2000 and SPECweb99. (The Xen team was quite helpful in providing details on the parameters they used for each test and even providing some of their testing scripts.) We generalized and extended their scripts into a test suite that would help save others this step in the future.

In our test suite, we replaced SPEC INT and SPECweb as the details of the test are not made public and they are only available for a fee from SPEC [SPECWEB] [SPECINT]. (Open benchmarks are nearly as important

as open source!) Instead of CPU-intensive SPECINT 2000, we chose FourInARow, an integer intensive program from freebench.org [FourInARow]. We wrote our own replacement for the web server benchmark, SPECweb99, using Apache JMeter. We discuss our web benchmark in more detail in Section 3.

Our final hurdle was that our initial measurements showed much lower performance for native Linux than [Xen03]. In comparing the details of our configuration with the Xen team, we discovered that performance is much higher with SMP support disabled.

With those hurdles behind us, we successfully reproduced measurements from [Xen03] comparing the performance of XenoLinux and UML to native Linux. In Figure 1, we show the results from Figure 3 of [Xen03] and our results. In Figure 1a, we show data from Figure 3 of [Xen03] (minus the data on VMWare workstation). In Figure 1b, we show our results from performing similar experiments. The native Linux results are with SMP disabled in all tests.

We add error bars to illustrate standard deviation where we ran at least 5 tests of each benchmark. OSDB on UML gave errors in the majority of runs. We received only one score for OSDB-IR and no scores for OSBD-OLTP from all our tests. We are missing some measurements for UML. We investigated further, but were unable to determine a cause.

Reporting standard deviation adds important information about the reliability of a reported score. The standard deviation of most benchmarks is less than 1%. Dbench has a standard deviation of 14% and 18% for native Linux and XenoLinux respectively.

In our tests, the relative performance of XenoLinux and UML compared to native Linux is nearly identical to the performance reported in [Xen03] as shown in Figures 1a and 1b. Our CPU-intensive and web server benchmarks are not directly comparable to SPEC INT and SPECweb99, but accomplish a similar purpose and demonstrate similar relative performance.

In Table 1, we extract only the Xen bars from Figure 1 for the benchmarks that are directly comparable: Linux build time, dbench, OSDB-IR and OSDB-OLTP. Our tests show Xen to be slightly slower relative to native Linux on three of the four repeated tests. In each case the difference is less than 5%, but it is also outside the standard deviation that we measured. Because the difference is so small in this case, we don't see a problem with the results in [Xen03]. However, it is a good illustration of the value of repeated results in validating published numbers.

Our web server benchmark shows Xen to be better than native Linux with SMP disabled. However, if we compare to Linux with SMP enabled, Xen and native Linux are nearly matched as shown in [Xen03] Figure 2. This is one frustration we had with the results in [Xen03]: some results are reported with SMP enabled and some with SMP disabled. The authors gave native Linux as much advantage as possible relative to Xen each time. This is certainly honorable, but it made repeating the results more difficult.

Finally, we are ready to answer our first question: Can we reproduce the results from the SOSP-03 Xen paper? We have mentioned a few caveats, but overall the answer is yes. We can reproduce the comparison of XenoLinux and native Linux to within a few percent on nearly identical hardware.

Performance Relative to Native Linux	Xen Repeated (std deviation)	Xen SOSP-03
Linux Build	0.943 (0.003)	0.970
OSDB-IR	0.892 (0.024)	0.919
OSDP-OLTP	0.905 (0.020)	0.953
dbench	0.962 (0.182)	0.957

Table 1 Comparing the Relative Performance of XenoLinux to native Linux in our repeated experiments to the results in the [Xen03]. Numbers represent the percentage of the original Linux performance retained in XenoLinux. Numbers in parentheses are the standard deviation.

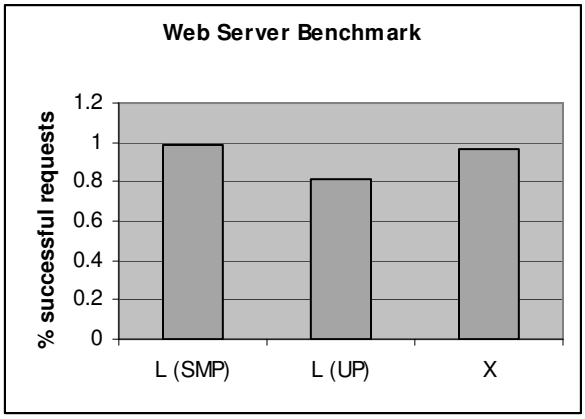


Figure 2 Comparing Xen to native Linux with SMP enabled to native Linux with SMP disabled for our web server benchmark.

3. Xen and Virtual Web Hosting

One of the stated goals of Xen is to enable applications such as server consolidation. In comparing Xen to Denali, [Xen03] page 2 states "Denali is designed to support thousands of virtual machines running network services which are small-scale and unpopular. In contrast, Xen is intended to scale to approximately 100 virtual machines running industry standard applications and services."

We set out to evaluate the suitability of Xen for virtual web hosting. Specifically, we wanted to determine how many usable guests could be supported for the purpose of hosting a web server.

[Xen03] includes a figure showing the performance of 128 guests each running CPU Intensive SPEC INT2000. We hoped to begin by showing the performance of 128 guests each running a web server benchmark. However, when we went to configure our Dell Xeon server for this test, we ran into certain resource limitations. First, as they state in the paper, the hypervisor does not support paging among guests to enforce resource isolation. Therefore each guest must have a dedicated region of memory. For the 128 guest SPEC INT test, they used 15 MB for each guest reserving 80 MB for the hypervisor and domain0 [Pratt03]. This is not sufficient for an industry standard web server. Second, they used raw disk partitions for each of the 128 guests. The Linux kernel supports only 15 total partitions per SCSI disk. Getting around this limit requires patching the kernel (as the Xen team did) or using a virtualized disk subsystem. We tried the virtualized disk subsystem in the Xen 1.0 source tree without success. We plan to evaluate the 1.1 source tree.

If we were to increase the memory allocated per guest from 15 MB to a more typical memory size of 128 MB, we could accommodate only 15 guests plus domain0. To support 100 guests at 128 MB per guest would require over 12 GB of memory. At 64 MB per guest, 100 guests would require over 6 GB of memory. In our Xeon server, the most memory we can support is 4 GB.

We also wished to re-evaluate the performance of multiple guests running concurrent web servers under load. Figure 4 of [Xen03] compares 1-16 concurrent web servers running as separate processes on native Linux to the same number running in their own Xen guest. The results indicate little degradation even at 16 concurrent servers.

Instead of using SPECweb99 to measure web server performance as in [Xen03], we wrote a replacement for it using Apache JMeter. JMeter is a flexible framework for testing functionality and performance of Web applications under load. More information including our JMeter test plans and documentation is available at http://www.clarkson.edu/class/cs644/xen.

Table 2 shows the type and distribution of requests sent to the Web servers under test in SPECweb99 [SPECWEB]. They base this distribution on an analysis of typical web server logs. We instrumented JMeter to follow the same distribution of requests and placed the proper static and dynamic content on each server.

SPECweb99 reports the number of simultaneous connections that meet a minimum set of error rate and bandwidth requirements [SPECWEB]. If a connection does not conform to the requirements, it does not contribute at all to the score. For our tests, we sent requests from JMeter engines on four remote clients. We do not factor out requests from "non-conforming" clients, nor do we limit the request rate from these machines. The tests complete after a specified number of requests have been issued. This number scales directly with the number of servers under test. We measured how many of the requests were returned correctly within 300 ms. We chose this value as a reasonable packet response time over a fast private LAN.

Due to the difference in reporting, we cannot compare SPECweb99 results directly to the results from our web server tests. Figure 3 reports our results for 1 to 16 concurrent servers. We report results for native Linux both with SMP enabled and disabled. For Xen, we allocated 98 MB for each guest in addition to domain0.

Our results show that native Linux with SMP enabled retains high performance even with 16 concurrent web server processes under high load significantly higher than SPECweb99. XenoLinux drops off steadily as more guests are added. Linux with SMP disabled is shown for completeness.

Thus, we are ready to answer our second question: Could we realistically use Xen for virtual web hosting? We have found Xen to be quite stable and could easily imagine using it for 16 moderately loaded servers. However, we would not expect to be able to support 100 guests running industry standard applications.

70% Static Content			
	35%	Less than 1KB class	9 files evenly distributed in the range
	50%	1-to-10-KB class	9 files evenly distributed in the range
	14%	10-to-100-KB class	9 files evenly distributed in the range
	1%	100KB-to-1MB class	9 files evenly distributed in the range
30% Dynamic Content			
	16%	POSTs to simulate user registration forms	Generic user registration form written.
	41.5%	GETs to simulate ad banner rotation	Number of banner files in SPECweb99 unknown; We used 3 files 468x60 pixels.
	42%	GETs with cookies	Set a single cookie < 1K.
	0.5%	CGI GETs for CGI web pages	Simple HTML page < 1K generated.

Table 2 Type and Distribution of Web Requests

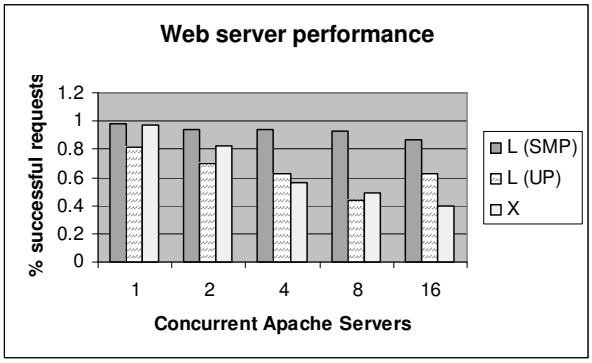


Figure 3 Web server performance for native Linux with SMP enabled, native Linux with SMP enabled and XenoLinux.

4. Comparing XenoLinux to Native Linux on Older PC Hardware

After reading [Xen03], we wondered how Xen would perform on an older PC rather than a new Xeon Server. So in addition to running on a 2.4 GHz dual processor server, we ran our tests on a P3 1 GHz processor with 512 MB of PC133 memory with 10/100 3COM (3c905C-TX/TX-M Ethernet card) and a 40 GB Western Digital WDC WD400BB-75AUA1 hard drive.

In Figure 4a, we first show the performance of Xen and native Linux on this older PC platform relative to native Linux on the Xeon server. Clearly raw performance is less on the older PC. In Figure 4b, we show the relative performance of Xen to native Linux on the older platform to the relative performance of Xen to native Linux on the faster platform. On average, Xen is only 3.5% slower relative to native Linux on the older PC.

Although the relative overhead is nearly the same on both systems, one disadvantage of the older PC is that we will be able to create fewer guests. For example, while we are able to create 16 guests with 128 MB of memory each on the Xeon server, we can create only 3 such guests plus domain0 on the older PC.

Thus, we are ready to answer our third question: Do you need $2500 Dell Xeon Server to run Xen effectively or will by 3 year old x86 do the job? No, an older PC can be used to efficiently use Xen, but only with a small number of guests.

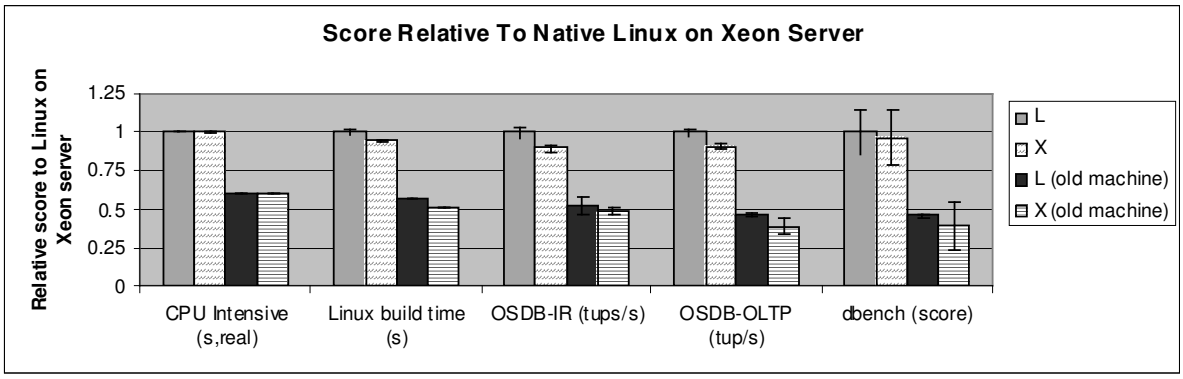


Figure 4a Relative Performance of native Linux and Xen on new Xeon server.

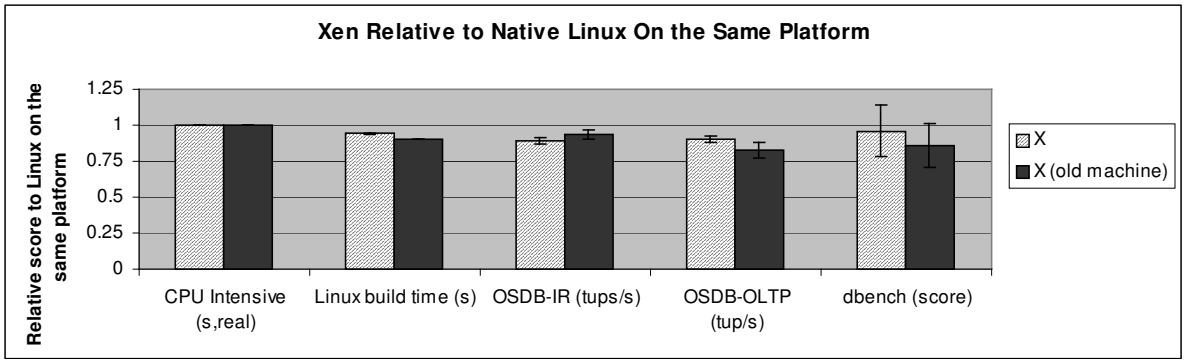


Figure 4b Relative Performance of Xen to native Linux on the same platform.

5. Xen on x86 vs IBM zServer

Virtualization for the x86 might be relatively new [Denali02, VMWare], but it has been around for over 30 years on IBM mainframes [VM370]. After reading [Xen03], it is natural to question how multiple Linux guests with Xen on x86 compare to multiple Linux guests on an IBM mainframe designed specifically to support virtualization. This is especially relevant given the following posting from Keir Fraser of the Xen team to the Linux Kernel Mailing List: "In fact, one of our main aims is to provide zseries-style virtualization on x86 hardware!" [LKML03]

In 2001, some of the authors won the top prize in the IBM Linux Challenge competition, a zServer. Specifically, we have an IBM eServer z800 model 2066-0LF with 1 processor and 8 GB of memory. It is connected to an ESS800 Enterprise Storage System via Ficon with 2 channel paths from 1 Ficon card. This machine was valued at over $200,000.

Our zServer is an entry-level model. The single CPU executes a dummy instruction every other cycle; a software upgrade is required to remove this feature. It could be configured to have up to 4 CPUs and up to 32 GB of memory. In addition, we could get up to 8 times the I/O bandwidth with additional FICON controllers

In Figure 5, we compare performance on the zServer to native Linux and Xen on both the new Xeon server and the old PC. On the zServer, we ran Linux guests with the 2.4.21 kernel just as in our x86 native Linux and Xen tests. For the zServer, it is specifically 2.4.21-1.1931.2.399.ent #1 SMP. We found that Xen on the Xeon server significantly outperforms the zServer on these benchmarks.

At first, we were surprised by these results. However, results presented by IBM in “Linux on zSeries Performance Update” by Thoss show comparable performance for a modern z900 with 1 CPU [ZPERF]. In Figure 6, we present a graph similar to one in [ZPERF, p14] showing the performance of dbench on our zServer, our Xeon server and our older x86. As in [ZPERF], throughput for a one CPU zServer does not rise above 150 MB/sec. However, we show a more significant degradation for more than 15 concurrent clients.

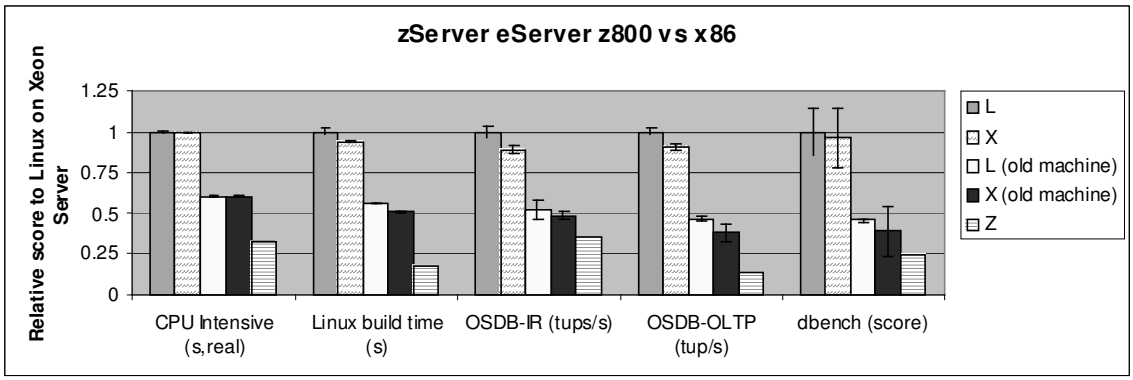


Figure 5 Performance on the zServer shown relative to native Linux on the Xeon server; Xen on the Xeon server as well as native Linux and Xen on the older PC also shown for comparison.

This comparison of our results to [ZPERF] leads us to believe that no simple software configuration enhancements will improve performance on our zServer, and that our figures although generated on an older model are comparable to more recent offerings from IBM. [ZPERF] also gives dbench scores for zServers with 4, 8 and 16 processors. Their results indicate that performance would be significantly better for a zServer with multiple processors. For example, [ZPERF] page 14 reports around 1000 MB/sec for a 16 CPU z900. We are also not testing all the features of the zSeries machines including high-availabilty, upgradability and manageability.

In Figure 7, we add measurements using our web server benchmark of the zServer with 1 to 16 Linux guests to the data presented in Figure 3. Xen on the Xeon server and the zServer perform similarly with the zServer performing better than Xen at 2, 4 and 16 guests, but worse at 1 and 8.

Thus, we are ready to answer our fourth question: How does a virtual machine monitor on commodity PCs compare to a virtual machine monitor on a mainframe? At least on our low-end zServer, Xen on x86 performs better for most workloads we examined. For a $2500 machine to do so well compared to a machine valued at over $200,000 is impressive!

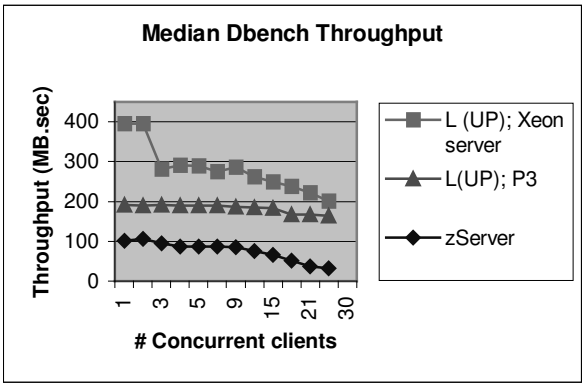


Figure 6 Throughput reported by dbench for 1 to 24 concurrent clients for native Linux on a Xeon server, native Linux on an older PC and Linux on the zServer.

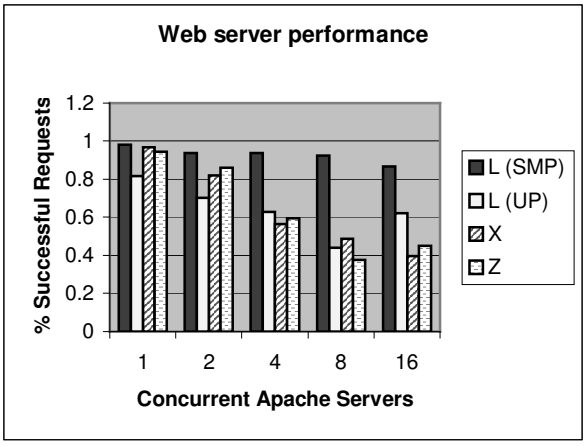


Figure 7 Web server performance on the zServer compared to native Linux with SMP enabled, native Linux with SMP enabled and XenoLinux on a Xeon server.

6. Experience With Related Research

The Xen team did a great job of facilitating repetition of their results, including releasing the code open source, producing a trial CD and responding happily to questions. Still, we were surprised to find how difficult it was to reproduce these results! It took a lot of investigation to assemble a comparable test platform and to reproduce the tests as run in [Xen03]. In the process, we ported three device drivers, wrote over a dozen testing scripts, wrote our own web server benchmark and ran hundreds of trials of each benchmark.

We make three main conclusions about repeated research. First, it is difficult enough that it should not be left as an exercise to the reader. Having another group repeat the initial results and polish some of the rough edges is important to the process of technology transfer. Second, it provides important validation of published results and can add additional insight beyond the original results. An independent third party is needed to verify the reliability of results reported, to question which tests are run, and to highlight some of the "spit and bailing wire" still present in the system. Finally, it is a great way to gain experience with research. This paper was a class project for an Advanced Operating Systems class at Clarkson University. This experience gave us a better appreciation for research in computer science than simply reading the other 30+ research papers in the class.

7. Conclusions

We were able to repeat the performance measurements of Xen published in "Xen and the Art of Virtualization" from SOSP-03. We find that Xen lives up to its claim of high performance virtualization of the x86 platform. We find that Xen can easily support 16 moderately loaded servers on a relatively inexpensive server class machine, but falls short of the 100 guest target they set. Xen performs well in tests on an older PC, although only a small number of guests could be supported on this platform. We find that Xen on x86 compares surprisingly well to an entry model zServer machine designed specifically for virtualization. We use this study as an example of repeated research and argue that this model of research, which is enabled by open source software, is an important step in transferring the results of computer science research into production environments.

8. Acknowledgements

We would like to thank the Xen group for releasing Xen as open source software. We would especially like to thank Ian Pratt, Keir Fraser and Steve Hand for answering our many emails. We'd also like to thank James Scott for help to port the driver for our SCSI controller. Thanks also to the Apache Foundation for JMeter. Thank you to our shepherd, Bart Massey for all of his detailed comments.

Many thanks to the members of the Clarkson Open Source Institute (COSI) and the Clarkson Internet Teaching Laboratory (ITL) for their patience as we tore apart lab machines to conduct our tests. Thanks also to Clarkson University and especially the Division of Mathematics and Computer Science for their support of the labs.

9. References

[AS3AP] C. Turbyfill, C. Orji, and D. Bitton. An ANSI SQL Standard Scalable and Portable Benchmark for Relational Database Systems. The Benchmark Handbook for Database and Transaction Processing, Chapter 4, pp 167-206.

[Denali02] A. Whitaker, M. Shaw, S. Gribble. Scale and Performance in the Denali Isolation Kernel. In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI 2002), ACM Operating Systems Review, Winter 2002 Special Issue, pages 195-210, Boston, MA, USA, December 2002.

[Disco97] E. Bugnion, S. Devine, K. Govil, M. Rosenblum. Disco: Running Commodity Operating Systems on Scalable Multiprocessors. ACM Transactions on Computer Systems, Vol. 15, No. 4, 1997, pp. 412-447.

[FourInARow] Freebench.org Project. URL http://www.freebench.org accessed December 2003.

[Google03] S. Ghemawat, H. Gobioff, S. Leung. The Google File System. Proceedings of the 19th ACM Symposium on Operating Systems Principles, 2003.

[JMETER] The Apache Jakarta Project. Jmeter. URL http://jakarta.apache.org/jmeter/index.html accessed December 2003.

[JMETER2] A. Bommarito, Regression Testing With JMeter.. URL http://www.phpbuilder.com/columns/bommarito20030610.php3 accessed December 2003.

[JMETER3] B. Kurniawan. Using JMeter. URL http://www.onjava.com/pub/a/onjava/2003/01/15/jmeter.html accessed December 2003.

[JMETER4] K. Hansen. Load Testing your Applications with Apache JMeter. URL http://javaboutique.internet.com/tutorials/JMeter accessed December 2003.

[LKML03] K. Fraser. Post to Linux Kernel Mailing List, October 3 2003, URL http://www.ussg.iu.edu/hypermail/linux/kernel/0310.0/0550.html accessed December 2003.

[LMBENCH] lmbench, URL http://www.bitmover.com/lmbench accessed December 2003.

[OSDB] Open source database benchmark system. URL http://osdb.sourceforge.net accessed December 2003.

[POSTGRES] M. Stonebraker. The Design Of The Postgres Storage System. Proceedings of the 13th Conference on Very Large Databases, Morgan Kaufman Publishers. (Los Altos CA), Brighton UK. 1987.

[POSTGRES2] PostgreSQL Global Development Group. URL http://www.postgresql.org accessed December 2003.

[Pratt03] Ian Pratt. Personal Communication. November 2003.

[SPECINT] CPU 2000, URL http://www.specbench.org/cpu2000 accessed December 2003.

[SPECWEB] SPECWEB99, URL http://www.spec.org/web99 accessed December 2003.

[UML01] Dike, Jeff. User-mode Linux. Proceedings of the 5th Annual Linux Showcase & Conference, Oakland CA (ALS 2001). pp 3-14, 2001.

[UML00] Dike, Jeff. A User-mode Port of the Linux Kernel. Proceedings of the 4 Annual Linux Showcase & Conference (ALS 2000), page 63, 2000.

[VM370] R. Creasy IBM Journal of Research and Development. Vol. 25, Number 5. Page 483. Published 1981. The Origin of the VM/370 Time-Sharing System.

[VMWARE] Vmware, URL http://www.vmware.com accessed December 2003.

[Voxel] Voxel Dot Net, URL http://www.voxel.net accessed December 2003.

[Xen99] D. Reed, I. Pratt, P. Menage, S. Early, and N. Stratford. Xenoservers: Accounted Execution of Untrusted Code. Proceedings of the 7th Workshop on Hot Topics in Operating Systems, 1999.

[Xen03] 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. Proceedings of the nineteenth ACM symposium on Operating systems principles, pp 164-177, Bolton Landing, NY, USA, 2003

[Xen03a] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, E. Kotsovinos, A. Madhavapeddy, R. Neugebauer, I. Pratt and A. Warfield. Xen 2002. Technical Report UCAM-CL-TR-553, January 2003.

[Xen03b] K. Fraser, S. Hand, T. Harris, I. Leslie, and I. Pratt. The Xenoserver Computing Infrastructure. Technical Report UCAM-CL-TR-552, University ofCambridge, Computer Laboratory, Jan. 2003.

[Xen03c] S. Hand, T. Harris, E. Kotsovinos, and I. Pratt. Controlling the XenoServer Open Platform, April 2003.

[Z800] IBM z800 Information, URL http://www-1.ibm.com/servers/eserver/zseries/800.html accessed December 2003.

[ZPERF] S. Thoss, Linux on zSeries Performance Update Session 9390. URL http://linuxvm.org/present/SHARE101/S9390a.pdf accessed December 2003.