Analysis of Virtualization Technologies for High Performance Computing Environments

Andrew J. Younge, Robert Henschel, James T. Brown, Gregor von Laszewski, Judy Qiu, Geoffrey C. Fox
Pervasive Technology Institute, Indiana University
2729 E 10th St., Bloomington, IN 47408, U.S.A.
{ajyounge,henschel,jatbrown,gvonlasz,xqiu,gcf}@indiana.edu

***Abstract*—As Cloud computing emerges as a dominant paradigm in distributed systems, it is important to fully understand the underlying technologies that make Clouds possible. One technology, and perhaps the most important, is virtualization. Recently virtualization, through the use of hypervisors, has become widely used and well understood by many. However, there are a large spread of different hypervisors, each with their own advantages and disadvantages. This paper provides an in-depth analysis of some of today's commonly accepted virtualization technologies from feature comparison to performance analysis, focusing on the applicability to High Performance Computing environments using FutureGrid resources. The results indicate virtualization sometimes introduces slight performance impacts depending on the hypervisor type, however the benefits of such technologies are profound and not all virtualization technologies are equal. From our experience, the KVM hypervisor is the optimal choice for supporting HPC applications within a Cloud infrastructure.**

I. Introduction

Cloud computing [1] is one of the most explosively expanding technologies in the computing industry today. A Cloud computing implementation typically enables users to migrate their data and computation to a remote location with some varying impact on system performance [2]. This provides a number of benefits which could not otherwise be achieved.

Such benefits include:

- *Scalability* - Clouds are designed to deliver as much computing power as any user needs. While in practice the underlying infrastructure is not infinite, the cloud resources are projected to ease the developer's dependence on any specific hardware.
- *Quality of Service (QoS)* - Unlike standard data centers and advanced computing resources, a well-designed Cloud can project a much higher QoS than traditionally possible. This is due to the lack of dependence on specific hardware, so any physical machine failures can be mitigated without the prerequisite user awareness.
- *Customization* - Within a Cloud, the user can utilize customized tools and services to meet their needs. This can be to utilize the latest library, toolkit, or to support legacy code within new infrastructure.
- *Cost Effectiveness* - Users find only the hardware required for each project. This reduces the risk for institutions potentially want build a scalable system, thus providing greater flexibility, since the user is only paying for needed infrastructure while maintaining the option to increase services as needed in the future.
- *Simplified Access Interfaces* - Whether using a specific application, a set of tools or Web services, Clouds provide access to a potentially vast amount of computing resources in an easy and user-centric way.

While Cloud computing has been driven from the start predominantly by the industry through Amazon [3], Google [4] and Microsoft [5], a shift is also occurring within the academic setting as well. Due to the many benefits, Cloud computing is becoming immersed in the area of High Performance Computing (HPC), specifically with the deployment of scientific clouds [6] and virtualized clusters [7].

There are a number of underlying technologies, services, and infrastructure-level configurations that make Cloud computing possible. One of the most important technologies is virtualization. Virtualization, in its simplest form, is a mechanism to abstract the hardware and system resources from a given Operating System. This is typically performed within a Cloud environment across a large set of servers using a Hypervisor or Virtual Machine Monitor (VMM), which lies in between the hardware and the OS. From the hypervisor, one or more virtualized OSs can be started concurrently as seen in Figure 1, leading to one of the key advantages of Cloud computing. This, along with the advent of multi-core processors, allows for a consolidation of resources within any data center. From the hypervisor level, Cloud computing middleware is deployed atop the virtualization technologies to exploit this capability to its maximum potential while still maintaining a given QoS and utility to users.

The rest of this paper is as follows: First, we look at what virtualization is, and what current technologies currently exist within the mainstream market. Next we discuss previous work related to virtualization and take an in-depth look at the features provided by each hypervisor. We follow this by outlining an experimental setup to evaluate a set of today's hypervisors on a novel Cloud test-bed architecture. Then, we look at performance benchmarks which help explain the utility of each hypervisor and the feasibility within an HPC environment. We conclude with our final thoughts and recommendations for using virtualization in Clouds for HPC.

II. Related Research

While the use of virtualization technologies has increased dramatically in the past few years, virtualization is not specific

978-0-7695-4460-1/11 $26.00 © 2011 IEEE
DOI 10.1109/CLOUD.2011.29

IEEE computer society

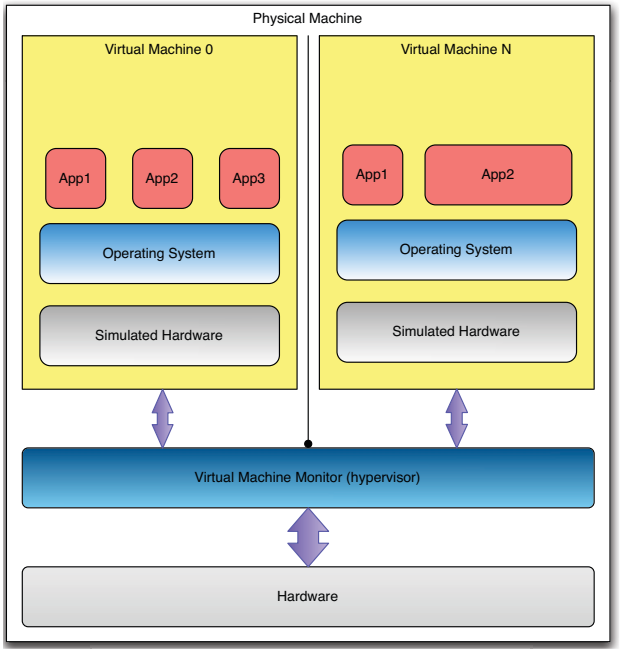

Fig. 1. Virtual Machine Abstraction

to the recent advent of Cloud computing. IBM originally pioneered the concept of virtualization in the 1960's with the M44/44X systems [8]. It has only recently been reintroduced for general use on x86 platforms. Today there are a number of public Clouds that offer IaaS through the use of virtualization technologies. The Amazon Elastic Compute Cloud (EC2) [9] is probably the most popular Cloud and is used extensively in the IT industry to this day. Nimbus [10], [11] and Eucalyptus [12] are popular private IaaS platforms in both the scientific and industrial communities. Nimbus, originating from the concept of deploying virtual workspaces on top of existing Grid infrastructure using Globus, has pioneered scientific Clouds since its inception. Eucalyptus has historically focused on providing an exact EC2 environment as a private cloud to enable users to build an EC2-like cloud using their own internal resources. Other scientific Cloud specific projects exist such as OpenNebula [13], In-VIGO [14], and Cluster-on-Demand [15], all of which leverage one or more hypervisors to provide computing infrastructure on demand. In recent history, OpenStack [16] has also come to light from a joint collaboration between NASA and Rackspace which also provide compute and storage resources in the form of a Cloud.

While there are currently a number of virtualization technologies available today, the virtualization technique of choice for most open platforms over the past 5 years has typically been the Xen hypervisor [17]. However more recently VMWare ESX [18] [1], Oracle VirtualBox [19] and the Kernel-based Virtual Machine (KVM) [20] are becoming more commonplace. As these look to be the most popular and feature-rich of al virtualization technologies, we look to evaluate all four to the fullest extent possible. There are however, numerious other virtualizaton technologies also available, including Microsoft's Hyper-V [21], Parallels Virtuozzo [22], QEMU [23], OpenVZ [24], Oracle VM [25], and many others. However, these virtualization technologies have yet to seen widespread deployment within the HPC community, at least in their current form, so they have been placed outside the scope of this work.

In recent history there have actually been a number of comparisons related to virtualization technologies and Clouds. The first performance analysis of various hypervisors started with, unsurprisingly, the hypervisor vendors themselves.

VMWare published their performance analysis in [26] as did the Xen developers in an their first paper [17]. The Xen paper compares Xen, XenoLinux, and VMWare across a number of SPEC and normalized benchmarks, resulting in a conflict between the two works. From here, a number of more unbiased reports originated, concentrating on server consolidation and web application performance [18], [27], [28] with fruitful yet sometimes incompatible results. A feature base survey on virtualization technologies [29] also illustrates the wide variety of hypervisors that currently exist. Furthermore, there has been some investigation into the performance within HPC, specifically with InfiniBand performance of Xen [30] and rather recently with a detailed look at the feasibility of the Amazon Elastic Compute cloud for HPC applications [31], however both works concentrate only on a single deployment rather than a true comparison of technologies.

As these underlying hypervisor and virtualization implementations have evolved rapidly in recent years along with virtualization support directly on standard x86 hardware, it is necessary to carefully and accurately evaluate the performance implications of each system. Hence, we conducted an investigation of several virtualization technologies, namely Xen, KVM, VirtualBox, and in part VMWare. Each hypervisor is compared alongside one another with bare-metal as a control and (with the exeption of VMWare) run through a number of High Performance benchmarking tools.

III. Feature Comparison

With the wide array of potential choices of virtualization technologies available, its often difficult for potential users to identify which platform is best suited for their needs. In order to simplify this task, we provide a detailed comparison chart between Xen 3.1, KVM from RHEL5, VirtualBox 3.2 and VMWWare ESX in Figure 2.

A. Virtualization methods

The first point of investigation is the virtualization method of each VM. Each hypervisor supports full virtualization, which is now common practice within most x86 virtualization deployments today. Xen, originating as a para-virtualized VMM, still supports both types, however full virtualization is often preferred as it does not require the manipulation of the guest kernel in any way. From the Host and Guest CPU lists, we see that x86 and, more specifically, x86-64/amd64 guests are all universally supported. Xen and KVM both suport

[1] Due to the restrictions in VMWare's licensing agreement, benchmark results are unavailable.

	Xen	KVM	VirtualBox	VMWare
Para-virtualization	Yes	No	No	No
Full virtualization	Yes	Yes	Yes	Yes
Host CPU	x86, x86-64, IA-64	x86, x86-64,IA64,PPC	x86, x86-64	x86, x86-64
Guest CPU	x86, x86-64, IA-64	x86, x86-64,IA64,PPC	x86, x86-64	x86, x86-64
Host OS	Linux, UNIX	Linux	Windows, Linux, UNIX	Proprietary UNIX
Guest OS	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX	Linux, Windows, UNIX
VT-x / AMD-v	Opt	Req	Opt	Opt
Cores supported	128	16	32	8
Memory supported	4TB	4TB	16GB	64GB
3D Acceleration	Xen-GL	VMGL	Open-GL	Open-GL, DirectX
Live Migration	Yes	Yes	Yes	Yes
License	GPL	GPL	GPL/proprietary	Proprietary

Fig. 2. A comparison chart between Xen, KVM, VirtualBox, and VMWare ESX

Itanium-64 architectures for full virtualization (due to both hypervisors dependency on QEMU), and KVM also claims support for some recent PowerPC architectures. However, we concern ourselves only with x86-64 features and performance, as other architectures are out of the scope of this paper. Of the x86-64 platforms, KVM is the only hypervisor to require either Intel VT-X or AMD-V instruction sets in order to operate. VirtualBox and VMWare have internal mechanisms to provide full virtualization even without the virtualization instruction sets, and Xen can default back to para-virtualized guests.

Next, we consider the host environments for each system. As Linux is the primary OS type of choice within HPC deployments, its key that all hypervisors support Linux as a guest OS, and also as a host OS. As VMWare ESX is meant to be a virtualization-only platform, it is built upon a specially configured Linux/UNIX proprietary OS specific to its needs. All other hypervisors support Linux as a host OS, with VirtualBox also supporting Windows, as it was traditionally targeted for desktop-based virtualization. However, as each hypervisor uses VT-X or AMD-V instructions, each can support any modern OS targeted for x86 platforms, including all variants of Linux, Windows, and UNIX.

While most hypervisors have desirable host and guest OS support, hardware support within a guest environment varies drastically. Within the HPC environment, virtual CPU (vCPU) and maximum VM memory are critical aspects to choosing the right virtualization technology. In this case, Xen is the first choice as it supports up to 128 vCPUs and can address 4TB of main memory in 64-bit modes, more than any other. VirtualBox, on the other hand, supports only 32 vCPUs and 16GB of addressable RAM per guest OS, which may lead to problems when looking to deploy it on large multicore systems. KVM also faces an issue with the number of vCPU supported limited to 16, recent reports indicate it is only a soft limit [32], so deploying KVM in an SMP environment may not be a significant hurdle. Furthermore, all hypervisors provide some 3D acceleration support (at least for OpenGL) and support live migration across homogeneous nodes, each with varying levels of success.

Another vital juxtaposition of these virtualization technologies is the license agreements for its applicability within HPC deployments. Xen, KVM, and VirtualBox are provided for free under the GNU Public License (GPL) version 2, so they are open to use and modification by anyone within the community, a key feature for many potential users. While VirtualBox is under GPL, it has recently also offered with additional features under a more proprietary license dictated by Oracle since its acquirement from Sun last year. VMWare, on the other hand, is completely proprietary with an extremely limited licensing scheme that even prevents the authors from willfully publishing any performance benchmark data without specific and prior approval. As such, we have neglected VMWare form the remainder of this paper. Whether going with a proprietary or open source hypervisor, support can be acquired (usually for an additional cost) with ease from each option.

B. Usability

While side by side feature comparison may provide crucial information about a potential user's choice of hypervisor, ease of installation and use are also important features that must be considered. We will take a look at each hypervisor from two user perspectives, a systems administrator and normal VM user.

One of the first things on any system administrator's mind on choosing a hypervisor is the installation. For all of these hypervisors, installation is relatively painless. The FutureGrid support group found that KVM and VirtualBox were the easiest to install, as there are a number of supported packages available and installation only requires the addition of one or more kernel modules and the support software. Xen, while still supported in binary form by many Linux distributions, is actually much more complicated. This is because Xen requires a full modification to the kernel itself, not just a module. Loading a new kernel into the boot process may complicate patching and updating later in the system's maintenance cycle. VMWare ESX, on the other hand, is entirely separate from most other installations. As previously noted, ESX is actually

a hypervisor and custom UNIX host OS combined, so installation of ESX is like installing any other OS from scratch. This may be either desirable or adverse, depending on the system administrator's usage of the systems and VMWare's ability to provide a secure and patched environment.

While system administrators may be concerned with installation and maintenance, VM users and Cloud developers are more concerned with daily usage. The first thing to note about all of such virtualiation technologies is they are supported (to some extent) by the libvirt API [33]. Libvirt is commonly used by many of today's IaaS Cloud offerings, including Nimbus, Eucalyptus, OpenNebula and OpenStack. As such, the choice of hypervisor for Cloud developer's is less of an issue, so long as the hypervisor supports the features they desire. For individual command line usage of each tool, it varies quite a bit more. Xen does provide their own set of tools for controlling and monitoring guests, and seem to work relatively well but do incur a slight learning curve. KVM also provides its own CLI interface, and while it is often considered less cumbersome it provides less advanced features directly to users, such as power management or quick memory adjustment (however this is subject to personal opinion). One advantage of KVM is each guest actually runs as a separate process within the host OS, making it easy for a user to manage and control the VM inside the host if KVM misbehaves. VirtualBox, on the other hand, provides the best command line and graphical user interface. The CLI, is especially well featured when compared to Xen and KVM as it provides clear, decisive and well documented commands, something most HPC users and system administrators alike will appreciate. VMWare provides a significantly enhanced GUI as well as a Web-based ActiveX client interface that allows users to easily operate the VMWare host remotely. In summary, there is a wide variance of interfaces provided by each hypervisor, however we recommend Cloud developers to utilize the libvirt API whenever possible.

IV. Experimental Design

In order to provide an unaltered and unbiased review of these virtualization technologies for Clouds, we need to outline a neutral testing environment. To make this possible, we have chosen to use FutureGrid as our virtualization and cloud test-bed.

A. The FutureGrid Project

FutureGrid (FG) [34] provides computing capabilities that enable researchers to tackle complex research challenges related to the use and security of Grids and Clouds. These include topics ranging from authentication, authorization, scheduling, virtualization, middleware design, interface design and cybersecurity, to the optimization of Grid-enabled and cloud-enabled computational schemes for researchers in astronomy, chemistry, biology, engineering, atmospheric science and epidemiology.

The test-bed includes a geographically distributed set of heterogeneous computing systems, a data management system that will hold both metadata and a growing library of software images necessary for Cloud computing, and a dedicated network allowing isolated, secure experiments, as seen in Figure 3. The test-bed supports virtual machine-based environments, as well as operating systems on native hardware for experiments aimed at minimizing overhead and maximizing performance. The project partners are integrating existing open-source software packages to create an easy-to-use software environment that supports the instantiation, execution and recording of grid and cloud computing experiments.

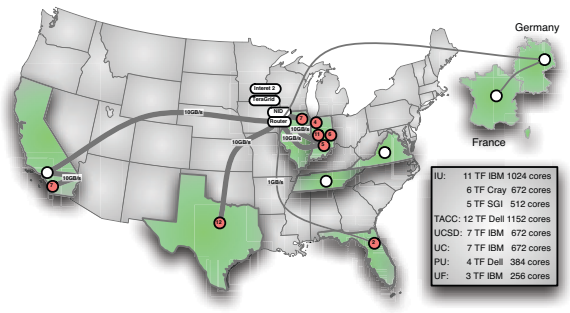

Fig. 3. FutureGrid Participants and Resources

One of the goals of the project is to understand the behavior and utility of Cloud computing approaches. However, it is not clear at this time which of these toolkits will become the users' primary choice. FG provides the ability to compare these frameworks with each other while considering real scientific applications [35]. Hence, researchers are able to measure the overhead of cloud technology by requesting linked experiments on both virtual and bare-metal systems, providing valuable information that help decide which infrastructure suits their needs and also helps users that want to transition from one environment to the other. These interests and research objectives make the FutureGrid project the perfect match for this work. During the initial conception of FutureGrid, Xen was selected as the default hypervisor as it was the best accepted hypervisor within the community at the time. However, we expect that the results gleaned from this paper will have a direct impact on the FutureGrid deployment itself as the performance of each selected hypervisor is evaluated.

B. Experimental Environment

Currently, one of FutureGrid's latest resources is the *India* system, a 256 CPU IBM iDataPlex machine consisting of 1024 cores, 2048 GB of ram, and 335 TB of storage within the Indiana University Data Center. Each compute node of India has two Intel Xeon 5570 quad core CPUs running at 2.93Ghz, 24GBs of Ram, and a QDR InfiniBand connection. A total of four nodes were allocated directly from India for these experiments. All were loaded with a fresh installation of Red Hat Enterprise Linux server 5.5 x86_64 with the 2.6.18-194.8.1.el5 kernel patched. Three of the four nodes were installed with different hypervisors; Xen version 3.1, KVM (build 83), and VirtualBox 3.2.10, and the fourth node was left as-is to act as a control for bare-metal native performance.

Each guest virtual machine was also built using Red Hat EL server 5.5 running an unmodified kernel using full virtualization techniques. All tests were conducted giving the guest VM 8 cores and 16GB of ram to properly span a compute node. Each benchmark was run a total of 20 times, with the results averaged to produce consistent results, unless indicated otherwise.

C. Benchmarking Setup

As this paper aims to objectively evaluate each virtualization technology from a side-by-side comparison as well as from a performance standpoint, the selection of benchmarking applications is critical.

The performance comparison of each virtual machine is based on two well known industry standard performance benchmark suites: HPCC and SPEC. These two benchmark environments are recognized for their standardized reproducible results in the HPC community, and the National Science Foundation (NSF), Department of Energy (DOE), and DARPA are all sponsors of the HPCC benchmarks. The following benchmarks provide a means to stress and compare processor, memory, inter-process communication, network, and overall performance and throughput of a system. These benchmarks were selected due to their importance to the HPC community since they are often directly correlated with overall application performance [36].

1) HPCC Benchmarks: The HPCC Benchmarks [37], [38] are an industry standard for performing benchmarks for HPC systems. The benchmarks are aimed at testing the system on multiple levels to test their performance. It consists of 7 different tests:

- *HPL* - The Linpack TPP benchmark measures the floating point rate of execution for solving a linear system of equations. This benchmark is perhaps the most important benchmark within HPC today, as it is the basis of evaluation for the Top 500 list [39].
- DGEMM - Measures the floating point rate of execution of double precision real matrix-matrix multiplication.
- *STREAM* - A simple synthetic benchmark program that measures sustainable memory bandwidth (in GB/s) and the corresponding computation rate for simple vector kernel.
- *PTRANS* - Parallel matrix transpose exercises the communications where pairs of processors communicate with each other simultaneously. It is a useful test of the total communications capacity of the network.
- *RandomAccess* - Measures the rate of integer random updates of memory (GUPS).
- *FFT* - Measures the floating point rate of execution of double precision complex one-dimensional Discrete Fourier Transform (DFT).
- *Communication bandwidth and latency* - A set of tests to measure latency and bandwidth of a number of simultaneous communication patterns; based on b_eff (effective bandwidth benchmark).

This benchmark suite uses each test to stress test the performance on multiple aspects of the system. It also provides reproducible results which can be verified by other vendors. This benchmark is used to create the Top 500 list [39] which is the list of the current top supercomputers in the world. The results that are obtained from these benchmarks provide an unbiased performance analysis of the hypervisors. Our results provide insight on inter-node PingPong bandwidth, PingPong latency, and FFT calculation performance.

2) SPEC Benchmarks: The Standard Performance Evaluation Corporation (SPEC) [40], [41] is the other major standard for evaluation of benchmarking systems. SPEC has several different testing components that can be utilized to benchmark a system. For our benchmarking comparison we will use the SPEC OMP2001 because it appears to represent a vast array of new and emerging parallel applications while simultaneously providing a comparison to other SPEC benchmarks. SPEC OMP continues the SPEC tradition of giving HPC users the most objective and representative benchmark suite for measuring the performance of SMP (shared memory multiprocessor) systems.

- The benchmarks are adapted from SPEC CPU2000 and contributions to its research program.
- The focus is to deliver systems performance to real scientific and engineering applications.
- The size and runtime reflect the needs of engineers and researchers to model large complex tasks.
- Two levels of workload characterize the performance of medium and large sized systems.
- Tools based on the SPEC CPU2000 toolset make these the easiest ever HPC tests to run.
- These benchmarks place heavy demands on systems and memory.

V. Performance Comparison

The goal of this paper is to effectively compare and contrast the various virtualization technologies, specifically for supporting HPC-based Clouds. The first set of results represent the performance of HPCC benchmarks. The mean value and standard deviation for each benchmark, represented by error bars, were calculated from the results of 20 runs. The benchmarking suite was built using the Intel 11.1 compiler, uses the Intel MPI and MKL runtime libraries, all set with defaults and no optimizations whatsoever.

We open first with High Performance Linpack (HPL), the de-facto standard for comparing resources. In Figure 4, we can see the comparison of Xen, KVM, and Virtual Box compared to native bare-metal performance. First, we see that native is capable of around 73.5 Gflops which, with no optimizations, achieves 75% of the theoretical peak performance. Xen, KVM and VirtualBox perform at 49.1, 51.8 and 51.3 Gflops, respectively when averaged over 20 runs. However Xen, unlike KVM and VirtualBox, has a high degree of variance between runs. This is an interesting phenomenon for two reasons. First, this may impact performance metrics for other HPC applications and cause errors and delays between even pleasingly-parallel applications and add to reducer function delays. Second, this wide variance breaks a key component of Cloud computing providing a specific and predefined quality of service. If

performance can sway as widely as what occurred for Linpack, then this may have a negative impact on users.

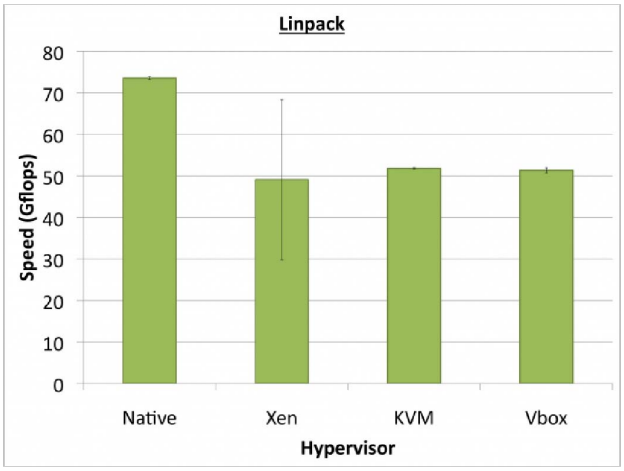

Fig. 4. Linpack performance

Next, we turn to another key benchmark within the HPC community, Fast Fourier Transforms (FFT). Unlike the synthetic Linpack benchmark, FFT is a specific, purposeful benchmark which provides results which are often regarded as more relative to a user's real-world application than HPL. From Figure 5, we can see rather distinct results from what was previously provided by HPL. Looking at Star and Single FFT, its clear performance across all hypervisors is roughly equal to bare-metal performance, a good indication that HPC applications may be well suited for use on VMs. The results for MPI FFT also show similar results, with the exception of Xen, which has a decreased performance and high variance as seen in the HPL benchmark. Our current hypothesis is that there is an adverse affect of using Intel's MPI runtime on Xen, however the investigation is still ongoing.

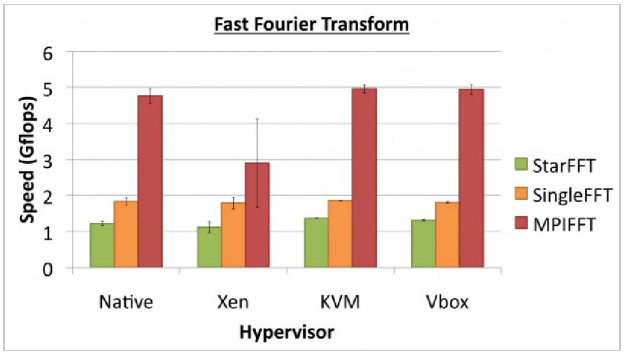

Fig. 5. Fast Fourier Transform performance

Another useful benchmark illustrative of real-world performance between bare-metal performance and various hypervisors are the ping-pong benchmarks. These benchmarks measure the bandwidth and latency of passing packets between multiple CPUs. With this experiment, all ping-pong latencies are kept within a given node, rather than over the network. This is done to provide further insight into the CPU and memory overhead withing each hypervisor. From Figure 6 the intranode bandwidth performance is uncovered, with some interesting distinctions between each hypervisor.

First, Xen performs close to native speeds on average, which is promising for the hypervisor. KVM, on the other hand, shows consistent overhead proportional to native performance across minimum, average, and maximum bandwidth. VirtualBox, on the other hand, performs well, in fact too well to the point that raises alarm. While the minimum and average bandwidths are within native performance, the maximum bandwidth reported by VirtualBox is significantly greater than native measurements, with a large variance. After careful examination, it appears this is due to how VirtualBox assigns its virtual CPUs. Instead of locking a virtual CPU to a real CPU, a switch may occur which could benefit on the off-chance the two CPU's in communication between a ping-pong test could in fact be the same physical CPU. The result would mean the ping-pong packet would remain in cache and result in a higher perceived bandwidth than normal. While this effect may be beneficial for this benchmark, it may only be an illusion towards the real performance gleaned from the VirtualBox hypervisor.

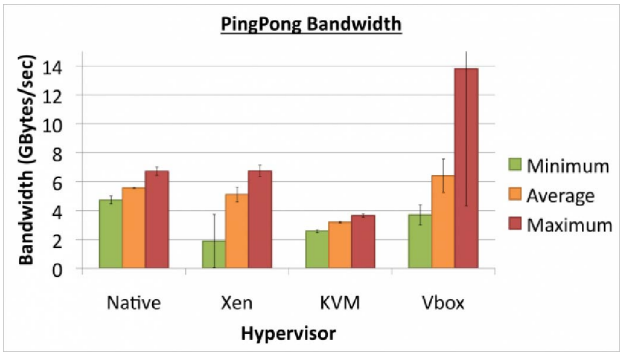

Fig. 6. Ping Pong bandwidth performance

The latency between each ping-pong is equally useful in understanding the performance impact of each virtualization technology. From Figure 7, we see KVM and VirtualBox have near-native performance; another promising result towards the utility of hypervisors within HPC systems. Xen, on the other hand, has extremely high latencies creating a high variance in the mean latency; the mean of its maximum latency was 3.4, which is off the chart but printed on the bar itself.

While the HPCC benchmarks provide a comprehensive view for many HPC applications including Linpack and FFT using MPI, performance of intra-node SMP applications using OpenMP is also investigated. Figure 8 illustrates SPEC OpenMP performance across our selected VMs, as well as baseline native performance. First, we see that the combined performance over all 11 applications executed 20 times yields the native testbed with the best performance at a SPEC score of 34465. KVM performance comes close with a score of 34384, which is so similar to the native performance that most users will never notice the difference. Xen and VirtualBox both perform notably slower with scores of 31824 and 31695,

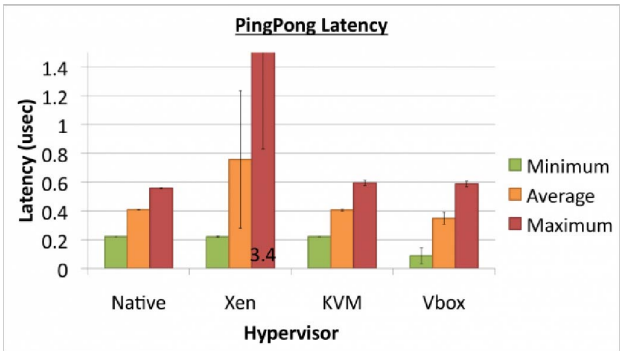

Fig. 7. Ping Pong latency performance (lower is better)

respectively, however this is only an 8% performance drop compared to native speeds.

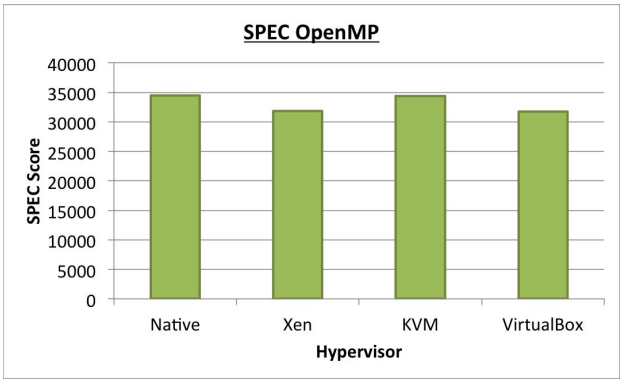

Fig. 8. Spec OpenMP performance

VI. Discussion

The primary goal of this paper is to evaluate the viability of virtualization within HPC. After our analysis, the answer seems to be a resounding ”yes.” However, we also hope to select the best virtualization technology for such an HPC environment. In order to do this, we combine the feature comparison along with the performance results, and evaluate the potential impact within the FutureGrid testbed.

From a feature standpoint, most of today's virtualization technologies fit the bill for at least small scale deployment, including VMWare. In short, each support Linux x86_64 platforms, use VT-X technology for full virtualization, and support live migration. Due to VMWare's limited and costly licensing, it is immediately out of competition for most HPC deployments. From a CPU and memory standpoint, Xen seems to provide the best expandability, supporting up to 128 cpus and 4TB of addressable RAM. So long as KVM's vCPU limit can be extended, it too shows promise as a feature-full virtualization technology. One of Virtualbox's greatest limitations was the 16GB maximum memory allotment for individual guest VMs, which actually limited us from giving VMs more memory for our performance benchmarks. If this can be fixed and Oracle does not move the product into the proprietary market, VirtualBox may also stand a chance for deployment in HPC environments.

	Xen	KVM	VirtualBox
Linpack	3	1	2
FFT	3	1	2
Bandwidth	2	3	1
Latency	3	2	1
OpenMP	2	1	3
Total Rating	13	8	9

Fig. 9. Benchmark rating summary (lower is better)

From the benchmark results previously described, the use of hypervisors within HPC-based Cloud deployments is mixed bag. Figure 9 summarizes the results based on a 1-3 rating, 1 being best and 3 being worst. While Linpack performance seems to take a significant performance impact across all hypervisors, the more practical FFT benchmarks seem to show little impact, a notably good sign for virtualization as a whole. The ping-pong bandwidth and latency benchmarks also seem to support this theory, with the exception of Xen, who's performance continually has wide fluctuations throughout the majority of the benchmarks. OpenMP performance through the SPEC OMP benchmarking suite also shows promising results for the use of hypervisors in general, with KVM taking a clear lead by almost matching native speeds.

While Xen is typically regarded as the most widely used hypervisor, especially within academic clouds and grids, it's performance lacks considerably when compared to either KVM or VirtualBox. In particular, Xen's wide and unexplained fluctuations in performance throughout the series of benchmarks suggests that Xen may not be the best choice for building a lasting quality of service infrastructure upon. From Figure 9, KVM rates the best across all performance benchmarks, making it the optimal choice for *general* deployment in an HPC environment.Furthermore, it may be possible to preemptively select the ideal Cloud environment for applications based on their similarity to the benchmarks presented in this paper and the variance we measure. We hope to further investigate this concept through the use of the FutureGrid experiment management framework at a later date.

In conclusion, it is the authors' projection that KVM is the best overall choice for use within HPC Cloud environments. KVM's feature-rich experience and near-native performance makes it a natural fit for deployment in an environment where usability and performance are paramount. Within the FutureGrid project specifically, we hope to deploy the KVM hypervisor across our Cloud platforms in the near future, as it offers clear benefits over the current Xen deployment. Furthermore, we expect these findings to be of great importance to other public and private Cloud deployments, as system utilization, Quality of Service, operating cost, and computational efficiency could all be improved through the careful evaluation of underlying virtualization technologies.

ACKNOWLEDGMENT

This document was developed with support from the National Science Foundation (NSF) under Grant No. 0910812 to Indiana University for "FutureGrid: An Experimental, High-Performance Grid Test-bed." Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF. We would also like to personally thank Shava Smallen, Greg Pike, Archit Kulshrestha, Fugang Wang, Javier Diaz, and the rest of the FutureGrid team for their continued help and support.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] L. Wang, G. von Laszewski, A. J. Younge, X. He, M. Kunze, and J. Tao, "Cloud Computing: a Perspective Study," *New Generation Computing*, vol. 28, pp. 63–69, Mar 2010. [Online]. Available: http://cyberaide.googlecode.com/svn/trunk/papers/08-lizhe-ngc/08-ngc.pdf

[3] "Amazon Elastic Compute Cloud." [Online]. Available: http://aws.amazon.com/ec2/

[4] E. Ciurana, *Developing with Google App Engine*. Springer, 2009.

[5] D. Chappell, "Introducing windows azure," Microsoft, Inc, Tech. Rep., 2009.

[6] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa, "Science clouds: Early experiences in cloud computing for scientific applications," *Cloud Computing and Applications*, vol. 2008, 2008.

[7] I. Foster, T. Freeman, K. Keahy, D. Scheftner, B. Sotomayer, and X. Zhang, "Virtual clusters for grid communities," *Cluster Computing and the Grid, IEEE International Symposium on*, vol. 0, pp. 513–520, 2006.

[8] R. Creasy, "The origin of the VM/370 time-sharing system," *IBM Journal of Research and Development*, vol. 25, no. 5, pp. 483–490, 1981.

[9] "Amazon elastic compute cloud," [Online], http://aws.amazon.com/ec2/.

[10] K. Keahey, I. Foster, T. Freeman, and X. Zhang, "Virtual workspaces: achieving quality of service and quality of life in the Grid," *Scientific Programming*, vol. 13, no. 4, pp. 265–275, 2005.

[11] K. Keahey, I. Foster, T. Freeman, X. Zhang, and D. Galron, "Virtual Workspaces in the Grid," *Lecture Notes in Computer Science*, vol. 3648, pp. 421–431, 2005. [Online]. Available: http://workspace.globus.org/papers/VW_EuroPar05.pdf

[12] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The Eucalyptus Open-source Cloud-computing System," *Proceedings of Cloud Computing and Its Applications*, 2008.

[13] J. Fontan, T. Vazquez, L. Gonzalez, R. S. Montero, and I. M. Llorente, "OpenNEbula: The Open Source Virtual Machine Manager for Cluster Computing," in *Open Source Grid and Cluster Software Conference*, San Francisco, CA, USA, May 2008.

[14] S. Adabala, V. Chadha, P. Chawla, R. Figueiredo, J. Fortes, I. Krsul, A. Matsunaga, M. Tsugawa, J. Zhang, M. Zhao, L. Zhu, and X. Zhu, "From virtualized resources to virtual computing Grids: the In-VIGO system," *Future Generation Comp. Syst.*, vol. 21, no. 6, pp. 896–909, 2005.

[15] J. Chase, D. Irwin, L. Grit, J. Moore, and S. Sprenkle, "Dynamic virtual clusters in a grid site manager," in *12th IEEE International Symposium on High Performance Distributed Computing, 2003. Proceedings*, 2003, pp. 90–100.

[16] Rackspace, "Openstack," WebPage, Jan 2011. [Online]. Available: http://www.openstack.org/

[17] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, New York, U. S. A., Oct. 2003, pp. 164–177.

[18] P. Padala, X. Zhu, Z. Wang, S. Singhal, and K. Shin, "Performance evaluation of virtualization technologies for server consolidation," HP Laboratories, Tech. Rep., 2007.

[19] J. Watson, "Virtualbox: bits and bytes masquerading as machines," *Linux Journal*, vol. 2008, no. 166, p. 1, 2008.

[20] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "kvm: the Linux virtual machine monitor," in *Proceedings of the Linux Symposium*, vol. 1, 2007, pp. 225–230.

[21] D. Leinenbach and T. Santen, "Verifying the Microsoft Hyper-V Hypervisor with VCC," *FM 2009: Formal Methods*, pp. 806–809, 2009.

[22] I. Parallels, "An introduction to os virtualization and parallels virtuozzo containers," Parallels, Inc, Tech. Rep., 2010. [Online]. Available: http://www.parallels.com/r/pdf/wp/pvc/Parallels_Virtuozzo_Containers_WP_an_introduction_to_os_EN.pdf

[23] D. Bartholomew, "Qemu: a multihost, multitarget emulator," *Linux Journal*, vol. 2006, no. 145, p. 3, 2006.

[24] 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," in *Proceedings of the 2007 workshop on Experimental computer science*, ser. ExpCS '07. New York, NY, USA: ACM, 2007.

[25] Oracle, "Performance evaluation of oracle vm server virtualization software," Oracle, Whitepaper, 2008. [Online]. Available: http://www.oracle.com/us/technologies/virtualization/oraclevm/026997.pdf

[26] K. Adams and O. Agesen, "A comparison of software and hardware techniques for x86 virtualization," in *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*. ACM, 2006, pp. 2–13, vMware.

[27] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu, "An analysis of performance interference effects in virtual environments," in *Performance Analysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on*. IEEE, 2007, pp. 200–209.

[28] S. Rixner, "Network virtualization: Breaking the performance barrier," *Queue*, vol. 6, no. 1, p. 36, 2008.

[29] S. Nanda and T. Chiueh, "A survey of virtualization technologies," Tech. Rep., 2005.

[30] A. Ranadive, M. Kesavan, A. Gavrilovska, and K. Schwan, "Performance implications of virtualizing multicore cluster machines," in *Proceedings of the 2nd workshop on System-level virtualization for high performance computing*. ACM, 2008, pp. 1–8.

[31] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. Wasserman, and N. Wright, "Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud," in *2nd IEEE International Conference on Cloud Computing Technology and Science*. IEEE, 2010, pp. 159–168.

[32] R. Harper and K. Rister, "Kvm limits arbitrary or architectural?" IBM Linux Technology Center, Jun 2009.

[33] M. Bolte, M. Sievers, G. Birkenheuer, O. Niehorster, and A. Brinkmann, "Non-intrusive virtualization management using libvirt," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, 2010, pp. 574 –579.

[34] "FutureGrid," Web Page, 2009. [Online]. Available: http://www.futuregrid.org

[35] G. von Laszewski, G. C. Fox, F. Wang, A. J. Younge, A. Kulshrestha, and G. Pike, "Design of the FutureGrid Experiment Management Framework," in *Proceedings of Gateway Computing Environments 2010 at Supercomputing 2010*, Nov 2010. [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/vonLaszewski-10-FG-exp-GCE10.pdf

[36] J. J. Dujmovic and I. Dujmovic, "Evolution and evaluation of spec benchmarks," *SIGMETRICS Perform. Eval. Rev.*, vol. 26, no. 3, pp. 2–9, 1998.

[37] P. Luszczek, D. Bailey, J. Dongarra, J. Kepner, R. Lucas, R. Rabenseifner, and D. Takahashi, "The HPC Challenge (HPCC) benchmark suite," in *SC06 Conference Tutorial*. Citeseer, 2006.

[38] J. Dongarra and P. Luszczek, "Reducing the time to tune parallel dense linear algebra routines with partial execution and performance modelling," niversity of Tennessee Computer Science Technical Report, Tech. Rep., 2010.

[39] J. Dongarra, H. Meuer, and E. Strohmaier, "Top 500 supercomputers," website, November 2008.

[40] K. Dixit, "The SPEC benchmarks," *Parallel Computing*, vol. 17, no. 10-11, pp. 1195–1209, 1991.

[41] SPEC, "Standard performance evaluation corporation," Webpage, Jan 2011. [Online]. Available: http://www.spec.org/