Evaluating Interconnect and Virtualization Performance for High Performance Computing

Lavanya Ramakrishnan R. Shane Canon Krishna Muriki
Iwona Sakrejda Nicholas J. Wright
NERSC, CRD, IT, Lawrence Berkeley National Lab
Berkeley, CA 94720
{lramakrishnan,scanon,kmuriki,isakrejda,njwright}@lbl.gov

ABSTRACT

Scientists are increasingly considering cloud computing platforms to satisfy their computational needs. Previous work has shown that virtualized cloud environments can have significant performance impact. However there is still a limited understanding of the nature of overheads and the type of applications that might do well in these environments. In this paper we detail benchmarking results that characterize the virtualization overhead and its impact on performance. We also examine the performance of various interconnect technologies with a view to understanding the performance impacts of various choices. Our results show that virtualization can have a significant impact upon performance, with at least a 60% performance penalty. We also show that less capable interconnect technologies can have a significant impact upon performance of typical HPC applications. We also evaluate the performance of the Amazon Cluster compute instance and show that it performs approximately equivalently to a 10G Ethernet cluster at low core counts.

1. INTRODUCTION

Cloud computing is a business and operations model for a large data center that allows for cost savings due to economies of scale. Today, scientists are increasingly considering cloud computing as a resource platform for their computing needs. Principly because of the previously mentioned potential cost savings when compared to alternatives such as purchasing their own machines.

The Infrastructure-as-a-Service (IaaS) type of cloud computing environment today typically consists of a group of machines connected over an ethernet connection, often running some kind of virtualization software. Amazon EC2 is a leading commercial example of this type of cloud service that is available today. Potentially one of the most attractive features of such an environment for scientific computing is that it allows a user to fully define their own software environment. Scientists often have complex software dependencies and version needs that can become challenging to manage in shared high performance supercomputing centers. For example, in some cases it is important to be able to analyze new results using old compilers/ kernels/codes etc for the purposes of reproducibility. Virtualized cloud computing environments are attractive to these user groups as they give them the ability to manage and control the software stack, resulting in software portability and productivity gains.

Early studies have benchmarked scientific applications on commercial platforms [13, 3, 12, 15, 17, 8]. These studies show that tightly coupled applications perform poorly in virtualized environments such as Amazon EC2. However there is limited understanding about the type of cloud environments (e.g., inteconnects, machine configuration etc) that might benefit science applications. HPC resource providers are interested in understanding the per-

formance tradeoffs of using cloud platform solutions such as virtualization in supercomputing centers. The Magellan project, funded by the DOE, is evaluating cloud solutions to understand their suitability for scientific computing.

Today's cloud environments are typically based on commodity hardware and use TCP over Ethernet to connect the nodes. However, compute resources found at HPC centers typically have much higher performing interconnects, such as InfiniBand, that provide higher bandwidth and lower latency. Thus the networking performance of current cloud environments is quite different from that at HPC centers. In this paper we use the Magellan cloud testbed to understand the performance impact of the fabrics (Infiniband, 10G) and protocols (Infiniband, TCP, virtualization) between an HPC resource and a typical cloud resource. Specifically we consider the following: native performance using InfiniBand, TCP over InfiniBand, TCP over 10 G Ethernet, TCP over 10G Ethernet with Virtualization and TCP over 1G Ethernet . Additionally, we compare the performance of Amazon Cluster Compute instance types, the specialized HPC offering over 10G Ethernet.

In this paper, we address the question of what must virtualized cloud environments provide in order to be beneficial for HPC applications. We use standard benchmarks such as the HPC-Challenge [7] to understand the overheads. We also select a subset of benchmarks from the NERSC6 benchmark suite based on our earlier work to capture the application behavior in virtualized environments. [8] Specifically, we make the following contributions in this paper,

- We evaluate the performance impact of each of the layers in the hierarchy of protocols encountered by scientific applications, i.e., InfiniBand, TCP over InfiniBand, TCP over Ethernet and TCP over 10 G and 1G Ethernet. Our results show that the performance differences between the interconnects is increased in situations where there is a lot of network contention, with many nodes simultaneously communicating, which leads to increased time to solution for scientific applications.

- We show that using virtualization has significant impacts upon application performance. For example, at the lowest count the virtualization layer causes a 1.6× and 2.5× performance decrease for PARATEC and MILC respectively compared to the same hardware configuration without using virtualization.

- We evaluate the performance of Amazon Cluster Compute instances, the specialized HPC offering. Our results show that at low core counts the performance is comparable to a 10G Ethernet cluster. As the core count increases and the network contention increases, the results drop below those of our 10G cluster. At the highest core count the performance for our PARATEC benchmark is decreased by a factor of 1.7×, presumably either due to increased virtualization overhead or an inferior network configuration.

Our paper is structured a follows: Section 2 describes related work. Section 3 provides an overview of our methodology. Section 4 details our performance analysis and discussion. We present our conclusions in Section 5.

2. RELATED WORK

Several groups have looked at the feasibility of cloud computing. However there has been no previous work analyzing the performance of interconnects in today's high performance computing clusters and cloud environments. Here we summarize related work that looks at the performance of cloud computing.

Various groups have evaluated cloud offering such as Amazon EC2 and hypervisors such as Xen and KVM. Previous work has focused on using standard benchmarks such as Linpack, NAS Parallel Benchmarks and other microbenchmarks [13, 3, 12, 15, 17, 8]. Performance of Xen and KVM environments for scientific applications has been studied [18, 4, 19, 14]

Application groups have looked at the performance and cost of porting specific application pipelines to Amazon EC2 cloud [16, 6, 1, 10, 11, 9].

In this paper we evaluate the hierarchy of protocols that impact the performance of the applications in cloud environments.

3. METHODOLOGY

Users typically use high performance supercomputing centers for their medium to large-sized runs. High performance supercomputing centers provide high performance networking and storage infrastructure, system administration and user support in addition to the compute servers that is beneficial to the users. However, in these environments

users have less flexibility and no explicit control of the software stack.

The recent emergence of cloud computing as a potential platform for scientific computing has resulted in the need to revisit scientific computing environments and consider the performance that is possible in these environments.Previous work has shown that virtualized cloud environments impact the performance of tightly coupled applications. However studies conducted on Amazon EC2 provide limited understanding of the causes of the performance decrease due to the blackbox nature of these cloud services. Thus, we use the Magellan testbed to understand the impact of performance on applications with different networking protocols and fabrics.

InfiniBand interconnects are common in supercomputing centers due to its performance and scalability. The InfiniBand transport protocol provides a richer set of capabilities than TCP by leveraging the underlying link layer protocol and RDMA features. We consider this as a baseline for performance benchmarking and quantify the loss in performance from each different configuration. Specifically we compare the following configurations a) Infiniband (RDMA) over Infiniband b) TCP over Infiniband c) TCP over 10G Ethernet d) Amazon Cluster Compute e) TCP over 10G Ethernet in Virtual machines and f) TCP over 1G Ethernet. We use standard benchmarks such as HPC-Challenge (HPCC) [7] and application benchmarks to understand how communication protocols impact performance and how they compare with the performance in virtualized environments. This approach enables us to determine the performance impact of the different protocols separate from the overhead from virtualization.

3.1 Machine Description

We use the Magellan systems and the Amazon Cluster Compute Instances in our experiments.

3.1.1 Amazon Cluster Compute

Amazon is a virtual computing environment that provides a web services API for launching and managing virtual machine instances. Amazon recently started providing access to Cluster Compute (CC) instances in addition to the other types of instances previously available. These new CC instances are significantly different from the other types in terms of performance characteristics. This is the first instance type which guarantees the hardware architecture of the nodes and reduces the chance for any performance variation due to varying hardware types. Amazon has defined the specification on these CC instances with dual socket Intel Nehelam CPUs at 2.97 GHz and 23 GB of memory per node. The nodes are connected by a 10G Ethernet network allowing applications on the nodes to communicate at high speed. Also Amazon guarantees that the hardware under CC instances is not shared with any other Amazon EC2 instances and at any given time each node will be running only one virtual instance (single occupancy). Another new feature with CC instances is a logical entity called Placement Group which helps in launching multiple CC instances into a cluster of instances with low communication latency by trying to place them as close as possible within the network. As of today Cluster Compute instances are available only in US East region and all instances are 64 bit and use a Linux operating system.

The Amazon environment provides a set of virtual machines with no shared resources between them. Almost all HPC applications assume the presence of a shared parallel filesystem between compute nodes, and a head node that can submit MPI jobs to all of the worker nodes. Thus we replicated a typical HPC cluster environment in the cloud by creating virtual clusters [5, 10]. We used a series of Python scripts to configure a file server, a head node, and a series of worker nodes. The head node could submit MPI jobs to all of the worker nodes, and the file server provided a shared filesystem between the nodes.

To implement the shared filesystem, we attached an Amazon Elastic Block Store (EBS) [2] device to the fileserver virtual machine. EBS provides a block level storage volume to EC2 instances that persists independently from the instance lifetimes. On top of the EBS volume we built a standard Linux ext3 file system, that was then exported via NFS to all of the virtual cluster nodes.

3.1.2 Magellan

All the experiments were performed using the Magellan compute resources at NERSC. Magellan is a 720 node IBM iDataPlex cluster. Each node has two quad-core Intel Nehalem processors running at 2.67 Ghz, 24GB of RAM and two network connections: a single Quad Data Rate (QDR) Infiniband network connection and a GiB ethernet connector. The IB network is locally a fat-tree with a global 2D-mesh. Our 10G network is based on the Juniper Qfabric Switching System with 4 Q/F nodes , 2 Q/F interconnects and Junos 11.3.

Our virtual machine environment is based on Eucalyptus 2.0, an open source software platform that allows organizations to leverage exisitng Linux-based infrastructure to create private clouds. Our Eucalyptus installation uses Kernel-based Virtual Machines(KVM) as a hypervisor. We modified Eucalyptus to use virtio for disk access. We use the KVM option for the emulated e1000 nic for the network. All codes were compiled with the Intel compiler version 11.1 and used version 1.4.2 of Open-MPI.

All tests were performed on an instance type configured with 8CPUs/20G memory/20 G disk. The guest OS is CentOS release 5.5 (Linux kernel 2.6.28-11). We setup a virtual cluster where the head node mounts a block store volume that has all the application binaries and data. The block store volume is mounted on the other virtual machines via NFS. All the virtual machine communication traffic goes over the Ethernet network.

In all cases we ran the benchmarks three times and report the best result. In all cases, the three measurements were in close agreement with each other.

3.2 Applications

In this work we used two codes from the NERSC6 benchmark suite, PARATEC and MILC. We chose these applications based upon our previous work which showed that they have differing communication characteristics, and therefore place different demands upon the computational resources. [8] In combination these two applications are representative of more than 20% of the NERSC workload.

MILC This code represents Lattice Computation that is used to study Quantum ChromoDynamics (QCD), the theory of the sub-atomic "strong" interactions responsible for binding quarks into protons and neutrons and holding them together in the nucleus. QCD discretizes space and evaluates field variables on sites and links of a regular hypercube lattice in four-dimensional space time. It involves integrating an equation of motion for hundreds or thousands of time steps that requires inverting a large, sparse matrix at each integration step. The sparse, nearly-singular matrix problem is solved using a conjugate gradient (CG) method and many CG iterations are required for convergence. Within a processor, the four-dimensional nature of the problem requires gathers from widely separated locations in memory. The inversion by CG requires repeated three-dimensional complex matrix-vector multiplications, which reduces to a dot product of three pairs of three-dimensional complex vectors. Each dot product consists of five multiply-add operations and one multiply. The parallel programming model for MILC is a 4-D domain decomposition in which each task exchanges data with its eight nearest neighbors as well as participating in the all-reduce calls with very small payload as part of the CG algorithm. MILC is extremely dependent on memory bandwidth and prefetching and exhibits a high computational intensity.

In this work we use a $64\times32\times32\times72$ global lattice with 2 quark flavors, four trajectories and 15 steps per trajectory; this results in over 35,000 CG iterations per run.

PARATEC (PARAllel Total Energy Code) performs *ab initio* Density Functional Theory quantum-mechanical total energy calculations using pseudopotentials, a plane wave basis set and an all-band (unconstrained) conjugate gradient (CG) approach. Part of the calculation is carried out in Fourier space; custom parallel three-dimensional FFTs are used to transform the wavefunctions between real and Fourier space.

PARATEC uses MPI and parallelizes over grid points, thereby achieving a fine-grain level of parallelism. The real-space data layout of wave-functions is on a standard Cartesian grid. In general, the speed of the FFT dominates the runtime, since it stresses global communications bandwidth, though mostly point-to-point, using relatively short messages. Optimized system libraries (such Intel MKL or AMD ACML) are used for both BLAS3 and 1-D FFT; this results in high cache reuse and a high percentage of per-processor peak performance.

The benchmark used here is based on the NERSC-6 input. The input contains 686 Silicon atoms in a diamond lattice configuration and runs for 20 conjugate gradient iterations. benchmarking. A real science run might use 60 or more iterations.

3.3 HPCC

We also used the High Performance Computing Challenge (HPCC) benchmark suite [7] in addition to the application benchmarks previously described. Specifically, we uses the two measures of latency and bandwidth in the HPCC suite to understand the communication charactersitics of the different fabrics and protocols.

4. RESULTS AND DISCUSSION

In our evaluation a) we compare the performance

of different interconnect configurations, b) we measure the overhead of virtualization and, c) finally we compare the performance on Amazon Cluster Compute instances.

4.1 HPCC

Table 1 shows the latency and bandwidth for different interconnects and protocols at concurrencies 32 and 256. The table shows the results of three types of network measurements: ping-pong, random-ring and natural-ring. Ping-pong is a measure of point-to-point bandwidth whereas Random-Ring consists of each task simultaneously sending to a randomly selected partner whereas Natural-Ring sends messages to another partner in its natural order. The sequence pingpong, natural ring to random ring represents an increase in the amount of network contention and thus allows an understanding to be gained of how a particular interconnect will behave under increasing load as well as if particular interconnects cope worse under load than others. Figures 1, 2 and 3 shows the results graphically across all the core counts measured.

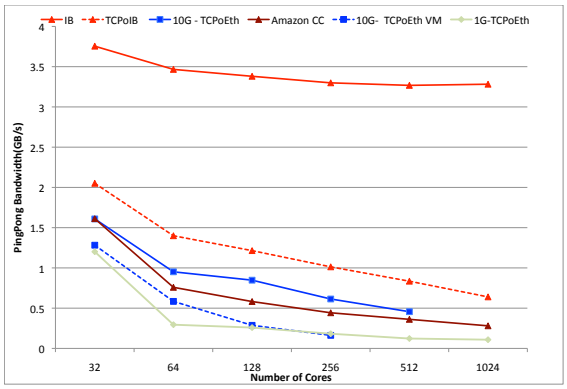
The ping-pong latency results at 32 cores shows that Infiniband has the lowest latency. The latency of 10G is about 6× that of IB. At higher concurrency (256) the 10GTCPoEth latency increases significantly showing almost a 5× increase from 32 cores. Infininband continues to show low latency at this concurrency while all the others show 2 to 2.5× increase in latency. The Amazon CC and the 10G - TCPoEth VM latency plots has a similar trend to the 10G - TCPoEth but show about 4 to 5 × increase in latency presumably from the virtualization overhead. The principle increase in latency occurs when switching to TCP over IB, then a further (much smaller) increase occurs with the different configurations. This indicates the latency is primarily a function of the transport mechanism, as one would expect. The Random Ring latency shows a 6 × increase in latency at the VM level from the 10GTCPoEth native at 32 cores, showing the impact of contention at the virtualization layer.

The Ping Pong BW of the TCPoIB connection is slightly better than 10G-TCPoEth at 32 cores but almost 2 × better at 256 cores. There is minimal impact from the virtualization on the Bandwidth at 32 cores but significant impact at 256 cores. The Random Ring Bandwidth clearly shows the lack of capability of the Ethernet connection to cope with significant amounts of network contention.

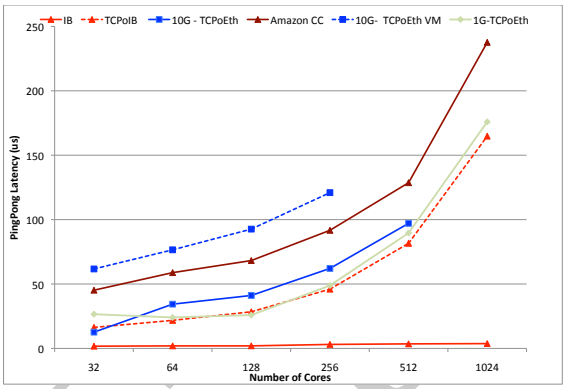
Generally speaking the results show that as the contention is increased the performance of all the interconnects decreases. It is also clear that the latencies are affected by contention by a greater amount than the bandwidths, and that as the core counts increase the decrease in the performance of the latencies is greater than that of the bandwidths. Thus not only do the more capable interconnects have measurably better performance at low core counts, their performance advantage increases as the core count and/or the amount of contention increases.

4.2 PARATEC and MILC performance

Figure 4 shows the performance of PARATEC and MILC for each of the different interconnect technologies and protocols. The performance for each is shown relative to the IB performance at the lowest core count. This allows us to show both the relative performance differences between the technologies as well as how they affect parallel scaling. Figure 4a shows that for PARATEC the Infiniband results are the fastest at all core counts. The change to protocol from IB to TCP, represented by the TCPoIB line, only affects performance at the higher core counts, 512 and 1024, where it makes the performance 1.5× and 2.5× slower than the native IB, respectively. Presumably this is because at these concurrencies the extra overhead required for TCP as compared to native IB communication is beginning to become important as at these concurrencies a larger quantity of smaller messages are being sent as compared to those at lower concurrencies. The 10G TCPoEth performance is within 10% of Infiniband at 32 cores but drops to about 2× slower at 512 cores. As expected 1G TCPoEth shows worse performance than the 10G. At 32 cores it is 1.5× slower than IB and at 256 cores it is about 3.5× slower. At 512 cores the runtime increases to more than 1.5× the 32 core value, indicating that the interconnect simply cannot keep up with the demands placed upon it by the application at this core count. The 10G TCPoEth VM results are by far the poorest. They show that the overhead of virtualization, at least as configured on the Magallen cluster, is significant. At 32 cores the performance is 1.75× worse than IB. As the core count increases the performance does not increase and the 10G VM line is not parallel to the 10G one, which indicates that the performance degradation due to virtualization is increasing as a function of core count. The Amazon CC results mirror the 10G TCPoEth ones at low core counts, which is a little surprising as the Amazon cluster also has virtualization. However the virtualization

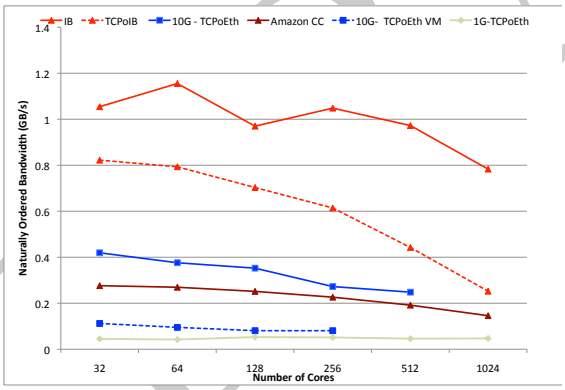


(a) PingPong Bandwidth

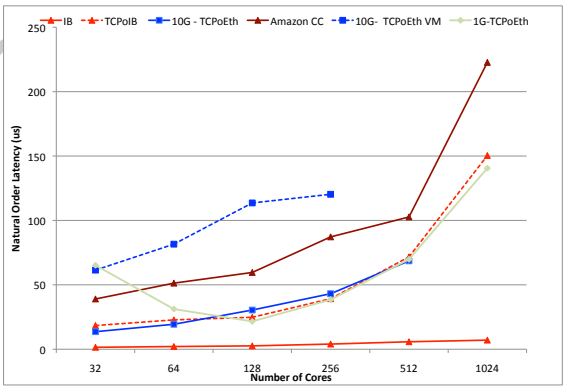


(b) PingPong Latency

Figure 1: Measurements of PingPong a) bandwidth and b) latency as a function of core count for various interconnect technologies and protocols.

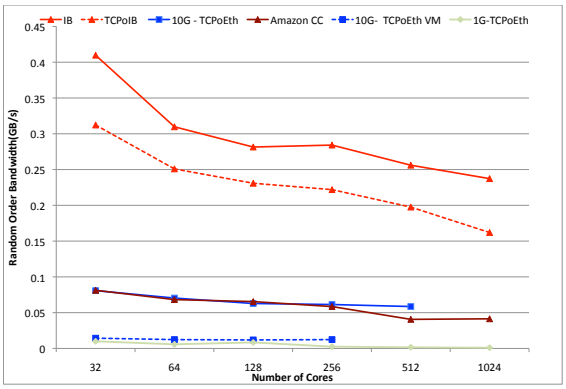


(a) Naturally Ordered Ring Bandwidth

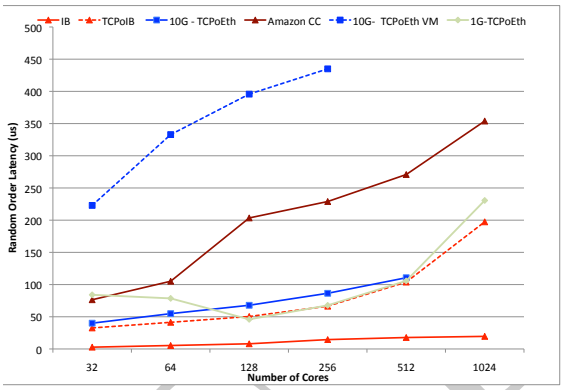


(b) Naturally Ordered Ring Latency

Figure 2: Measurements of Naturally ordered ring a) bandwidth and b) latency as a function of core count for various interconnect technologies and protocols

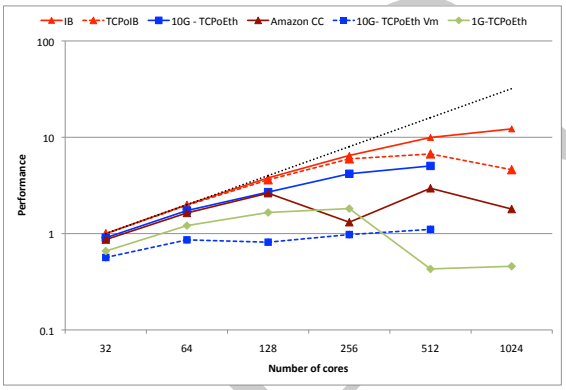


(a) Random Ring Bandwidth

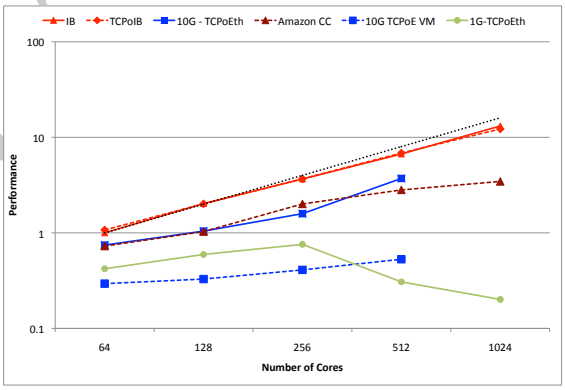


(b) RandomRing Latency

Figure 3: Measurements of Random Ring a) bandwidth and b) latency as a function of core count for various interconnect technologies and protocols



(a) PARATEC



(b) MILC

Figure 4: Performance of a) PARATEC and b) MILC plotted on a log-log scale as a function of core count using several different interconnect technologies and/or protocols. The dotted line represents ideal scaling based upon the IB performance at the lowest core count.

Table 1: HPCC Performance. All bandwidths are in GB/s and latencies in μs.

Cores	Fabric/Protocol	PingPongLat	NatOrLat	RandOrLat	PingPongBW	NatOrBW	RandOrBW
32	IB	1.72856	1.50204	2.79134	3.7545	1.05444	0.409803
32	TCPoIB	16.3001	18.4059	32.6339	2.0509	0.82211	0.312221
32	10G - TCPoEth	12.6114	13.6137	40.0417	1.60887	0.419289	0.0808118
32	Amazon CC	45.1947	39.0053	76.4316	1.6116	0.276396	0.0812501
32	10G- TCPoEth VM	61.6953	61.4166	222.948	1.28227	0.112029	0.0143262
32	1G-TCPoEth	26.5507	65.2075	84.1106	1.20135	0.0450691	0.00994543
256	IB	3.08863	4.00543	14.5425	3.29896	1.04822	0.284198
256	TCPoIB	45.994	39.196	66.4258	1.01308	0.614246	0.221942
256	10G - TCPoEth	62.0875	43.1061	86.4562	0.613839	0.272497	0.0613771
256	Amazon CC	91.6839	87.2135	228.95	0.442522	0.226378	0.0585243
256	10G- TCPoEth VM	120.933	120.306	434.962	0.162396	0.0806004	0.0123568
256	1G-TCPoEth	48.8237	38.5046	67.7118	0.182378	0.0512721	0.00251061

technologies underlying Amazon are different from that on the Magellan cluster. Additionally, Amazon likely has a much more highly tuned VM environment than what is available on the Magellan cluster through vanilla installation of open-source cloud software. At 256 cores and above the performance of the Amazon CC starts decreasing, as compared to the 10G TCPoEth one, presumably either because the performance of the 10G switch on the Magellan cluster is greater than that on the Amazon cluster or due to virtualization overheads at higher core counts similar to the 10G VMs on Magellan.

Figure 4b shows that for MILC, Infiniband is also the fastest at all core counts. In this case the TCPoIB results are indistinguishable from the native IB ones. Also, the performance at the highest core count, 1024, is still extremely good, indicating that we are still in the region where the application is scaling well, in contrast to the PARATEC case. The 10G TCPoEth is minimally 35% slower at all core counts. The performance of 1G TCPoEth is about 2.5× slower than IB at 64 cores and is about 4.8× slower at 256 cores. Again, above 256 cores the interconnect simply cannot keep up. The 10G TCPoEth VM results are by far the poorest. At 32 cores, the performance is 3.4× worse than the IB one, at 256 cores it is almost 9× worse. The Amazon CC numbers almost exactly mirror the 10G TCPoEth ones, in contrast the PARATEC case, again, because at these core counts where the MILC application is scaling better.

In both cases the performance differences between the interconnect technologies is smallest at low core counts, which makes sense as that is the point at which the applications are communicating the least. As the concurrency increases the differences between the performance for each interconnect type become greater because the applications are communicating more frequently, sending more data and thus stressing the interconnect more. This is identical to the trends we observed for the HPCC data, and is even more true for PARATEC than for MILC at these core counts.

Generally speaking HPC applications are run at the highest possible concurrency at which they run efficiently, in order to minimize the wallclock time to solution. These results demonstrate the productivity advantages that a scientist can gain by using a computing resource that has a high performance interconnect. For example, PARATEC using 256 cores is 1.5× faster using IB than 10GE. Thus a cluster that was 1.5× smaller in terms of number of nodes could be purchased to achieve the same throughput. Note that for the 1G TCPoEth the ratio is 3.5×.

We also note that the basic performance trends can be understood with reference to the HPCC data. Again, the greater the congestion and the core count the worse the low performing interconnects are.

5. CONCLUSIONS

Virtualized cloud computing environments promise to be useful for scientific application that need customized software environments. However, earlier studies have shown that virtualization has a significant performance impact for scientific applications. In this paper we analyzed the performance of a number of different interconnect technologies to understand the performance tradeoffs in this space and gain an understanding of what a private cloud configured for scientific computing should look like.

Our benchmarking approach enabled us to understand the impact of the layers in the hierarchy

of protocols. Although performance is highly dependent on the type of workload, we see that the performance decrease from virtualization is largely due to overheads in the communication, i.e., the reliance of the virtual machines on TCP over ethernet as the communication mechanism.

Our results show that while the differences between the interconnects is small at lower core counts, the impact is significant even at the midrange size of problems of 256 to 1024. While the bandwidth is slightly impacted at higher concurrency, the latency takes a much higher hit. Scientific applications tend to run at higher concurrencies to achieve the best time to solution. Thus to serve the needs of these applications, we need "good" interconnects.

In addition to the networking fabric and the protocol, the virtualization software stack imposes an additional overhead to the performance of the applications. This overhead also is impacted by the communication pattern of the application and the concurrency of the run.

The higher bound on the performance on virtual machines today is what is achievable with TCP over ethernet clusters. Our experiments show that the availability of InfiniBand interconnects on virtual machines would boost the performance of scientific applications by reducing the communication overheads. In that case scientific applications will be able to leverage the benefits of virtualization if future environments had InfiniBand available for the communication network. Thus, with InfiniBand on virtual machines we would be able to get a significant increase in performance. This is a topic we will explore in future work.

6. ACKNOWLEDGEMENTS

This work was funded in part by the Advanced Scientific Computing Research (ASCR) in the DOE Office of Science under contract number DE-C02-05CH11231. The authors would like to thank John Shalf, Keith Jackson, Harvey Wasserman, Shreyas Cholia, Jeff Broughton, Kathy Yelick for discussions.

7. REFERENCES

[1] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good. The cost of doing science on the cloud: the montage example. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, pages 1–12. IEEE Press, 2008.

[2] Amazon Elastic Block Store. `http://aws.amazon.com/ebs/`.

[3] C. Evangelinos and C. Hill. Cloud Computing for parallel Scientific HPC Applications: Feasibility of running Coupled Atmosphere-Ocean Climate Models on AmazonâĂŹs EC2. *ratio*, 2(2.40):2–34, 2008.

[4] M. Fenn, M. Murphy, and S. Goasguen. A study of a KVM-based cluster for grid computing. In *Proceedings of the 47th Annual Southeast Regional Conference*, pages 1–6. ACM, 2009.

[5] I. Foster, T. Freeman, K. Keahey, D. Scheftner, B. Sotomayor, and X. Zhang. Virtual clusters for grid communities. In *Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid*, pages 513–520. Citeseer, 2006.

[6] S. Hazelhurst. Scientific computing using virtual high-performance computing: a case study using the Amazon elastic computing cloud. In *Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology*, pages 94–103. ACM, 2008.

[7] HPCC benchmark web page: `http://icl.cs.utk.edu/hpcc/`.

[8] 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*, 2010.

[9] K. Keahey. Cloud Computing for Science. In *Proceedings of the 21st International Conference on Scientific and Statistical Database Management*, page 478. Springer-Verlag, 2009.

[10] 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*, 2008, 2008.

[11] K. Keahey, T. Freeman, J. Lauret, and D. Olson. Virtual workspaces for scientific applications. In *Journal of Physics: Conference Series*, volume 78, page 012038. Institute of Physics Publishing, 2007.

[12] R. Masud. High Performance Computing

with Clouds.

[13] J. Napper and P. Bientinesi. Can cloud computing reach the top500? In *Proceedings of the combined workshops on UnConventional high performance computing workshop plus memory access workshop*, pages 17–20. ACM, 2009.

[14] Nathan Regola and Jean Christophe Ducom. Recommendations for Virtualization Technologies in High Performance Computing. In *2nd IEEE International Conference on Cloud Computing Technology and Science*, 2010.

[15] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema. An early performance analysis of cloud computing services for scientific computing. *Delft University of Technology, Tech. Rep*, 2008.

[16] M. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel. Amazon S3 for science grids: a viable solution? In *Proceedings of the 2008 international workshop on Data-aware distributed computing*, pages 55–64. ACM, 2008.

[17] J. Rehr, F. Vila, J. Gardner, L. Svec, and M. Prange. Scientific computing in the cloud. *Computing in Science and Engineering*, 99(PrePrints), 2010.

[18] G. Wang and T. E. Ng. The impact of virtualization on network performance of amazon ec2 data center. In *Proceedings of IEEE INFOCOM*, 2010.

[19] L. Youseff, R. Wolski, B. Gorda, and C. Krintz. Evaluating the performance impact of xen on mpi and process execution for hpc systems. In *Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed computing*, page 1. IEEE Computer Society, 2006.