SR-IOV: Performance Benefits for Virtualized Interconnects

Glenn K. Lockwood
San Diego Supercomputer Center
University of California, San Diego
La Jolla, CA 92093
glock@sdsc.edu

Mahidhar Tatineni
San Diego Supercomputer Center
University of California, San Diego
La Jolla, CA 92093
mahidhar@sdsc.edu

Rick Wagner
San Diego Supercomputer Center
University of California, San Diego
La Jolla, CA 92093
rpwagner@sdsc.edu

ABSTRACT

The demand for virtualization within high-performance computing is rapidly growing as new communities, driven by both new application stacks and new computing modalities, continue to grow and expand. While virtualization has traditionally come with significant penalties in I/O performance that have precluded its use in mainstream large-scale computing environments, new standards such as Single Root I/O Virtualization (SR-IOV) are emerging that promise to diminish the performance gap and make high-performance virtualization possible.

To this end, we have evaluated SR-IOV in the context of both virtualized InfiniBand and virtualized 10 gigabit Ethernet (GbE) using micro-benchmarks and real-world applications. We compare the performance of these interconnects on non-virtualized environments, Amazon's SR-IOV-enabled C3 instances, and our own SR-IOV-enabled InfiniBand cluster and show that SR-IOV significantly reduces the performance losses caused by virtualization. InfiniBand demonstrates less than 2% loss of bandwidth and less than 10% increase in latency when virtualized with SR-IOV. Ethernet also benefits, although less dramatically, when SR-IOV is enabled on Amazon's cloud.

Categories and Subject Descriptors

H.3.4 Performance evaluation, D.4.4 Network communication

General Terms

Performance, Experimentation

Keywords

Parallel computing, virtualization, cloud computing, infiniband

1. INTRODUCTION

As the reach of high-performance computing continues to extend beyond the traditional supercomputing applications dominated by mathematical and physical sciences[1], the demand for increased flexibility from high-performance cyberinfrastructure has been increasing. An increasing number of total jobs executed across XSEDE resources have been originating from web-based portals rather than the traditional command-line interfaces[13], and these science gateways are developing increasingly sophisticated software stacks that not only provide scientific applications, but external interfaces and services that lower the barrier for new users and communities to begin using XSEDE's advanced computational resources.

Maintaining resource reliability and increasing accessibility are concepts that often run orthogonally though; for example, the Neuroscience Gateway[16] portal allows users to submit raw C++ code to describe custom models to be run within the NEURON simulation environment[6]. While an undeniably useful feature for researchers, allowing users to submit arbitrary code from a web-based front-end to be executed on a HPC backend poses some security concerns. Thus, it becomes highly desirable to fence off portions of the underlying computing platform to mitigate the risks of allowing remote code execution, and accomplishing this by running all such codes from within a virtual machine (VM) is an appealing prospect. In addition, allowing users full control over their jobs' execution environment opens the opportunity for scientific communities to develop virtual "compute appliances" that contain tightly integrated application stacks that can be deployed in a hardware-agnostic fashion. These benefits have been a major driving force behind compute clouds such as Amazon Web Services (AWS) EC2.

Unfortunately, the input/output (I/O) performance of virtualized systems traditionally has been markedly worse than that of the native hardware[12], and this performance disparity has severely limited the use of virtualization and cloud computing for scientific application[8]. However, hardware vendors have been providing an increased support for virtualization within hardware such as the I/O memory management unit (IOMMU)[21,23], and with the standardization and adoption of technologies such as Single-Root I/O Virtualization (SR-IOV) in I/O device hardware, a road has been paved towards truly high-performance virtualization for high-performance computing applications.

Given the increasing demand for the flexibility afforded by virtualization and the emergence of technologies enabling high-performance I/O from within VMs, the study presented in this manuscript aimed to develop a broad but quantitative evaluation of how these technologies might fit into the next generation of XSEDE resources. In particular, the performance of InfiniBand virtualized with SR-IOV was compared to native InfiniBand performance to determine if SR-IOV can bring the performance of virtual appliances and gateway-oriented VMs to levels

XSEDE '14, July 13 - 18 2014, Atlanta, GA, USA
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2893-7/14/07…$15.00.
http://dx.doi.org/10.1145/2616498.2616537

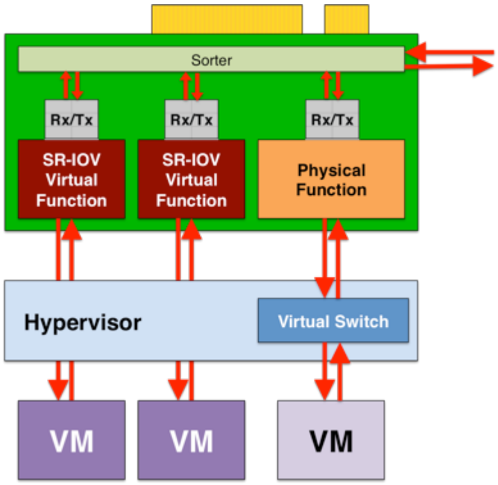

Figure 1. Data path for virtualized I/O with SR-IOV via Virtual Functions (VFs), and data path through the hypervisor via the Physical Function (PF).

comparable to native performance. In addition, performance analysis was also conducted on a widely used commercial cloud computing environment, Amazon EC2's C3 instances which use SR-IOV to virtualize its 10 gigabit Ethernet (10GbE) networking.

2. SINGLE ROOT I/O VIRTUALIZATION

The relatively poor I/O performance of virtual machines has long suffered because high-performance I/O is enabled by I/O devices' ability to perform DMA—direct memory access—whereby the I/O device can write directly to the compute host's memory without having to interrupt the host CPU. By bypassing the CPU entirely, DMA I/O operations are not subject to the thousands of cycles that the I/O stack of the host OS may impose for the operation. As a result, these I/O operations can occur with very low latency.

Within a VM, though, DMA is not as simple because the memory address space within the VM (and, by extension, the address to which the DMA operation should write) is not the same as the underlying host's real memory address space. Thus, while a VM can trigger a DMA operation, the VM's hypervisor needs to intercept the DMA and translate the operation's memory address from the VM's address space to the host's. As a result, DMAs originating within a VM still interrupt the host CPU so that the hypervisor can perform this address translation. The impact worsens if the I/O operation involves a network adapter, because if multiple VMs are running on the same host, the hypervisor must then also act as a virtual network switch. As one might expect, the more time the data takes to be processed through the hypervisor, the higher the latency for that I/O operation.

To address the address translation required by VMs, all major CPU vendors have incorporated the ability to do address translation for VMs into their existing IOMMUs, and this capability carries branding such as Intel(R) VT-d and AMD-Vi. These IOMMU technologies are what allow hosts to provide *PCIe passthrough* to VMs by acting as the interface between an I/O device on the PCIe bus and the VM's memory address space in lieu of the CPU-driven hypervisor. However, this support for virtualization within the IOMMU does not provide any virtual switching, and as such, is not sufficient to allow multiple VMs to efficiently share a single I/O device like an InfiniBand host channel adapter. In the case of multiple VMs on a single physical host, significant software logic is still required within the hypervisor's device driver to ensure that I/O operations requested by each VM are coalesced as efficiently as possible before they are passed to the I/O device to reduce excessive, latency-inducing interrupts.

Rather than address the problem of hardware-accelerated virtual switching, the PCI Special Interest Group drafted the Single Root I/O Virtualization (SR-IOV) specification[19], which outlines a standardized way for a single physical I/O device to present itself to the IOMMU (or more precisely, the PCIe bus) as multiple virtual devices. This allows the logic for coalescing I/O operations to reside in the I/O device itself, and the IOMMU can behave as if each virtual device is being presented to a single VM using PCIe passthrough.

These virtual devices, called *virtual PCIe functions* (VFs), are lightweight versions of the true *physical PCIe function* (PF). Although these VFs share most of the I/O device's underlying physical hardware, each VF has its own

1. PCIe route ID - thus, it really appears as a unique PCIe function on the bus
2. Configuration space, base address registers, and memory space
3. Send/receive queues (or *work queues*), complete with their own interrupts

These features allow the virtual functions to be interrupted independently of each other and process their own DMA streams (Figure 1). In addition, the I/O device still maintains its fully featured PF that can interact with the hypervisor. The actual logic

Table 1. Summary of tested platforms

	Native InfiniBand (SDSC)	SR-IOV InfiniBand (SDSC)	Native 10GbE (SDSC)	Software-Virtualized 10GbE (AWS)	SR-IOV 10GbE (AWS)
Platform	Rocks 6.1 (EL6)	Rocks 6.1 (EL6) kvm Hypervisor	Rocks 6.1 (EL6)	Amazon Linux 2013.09 (EL6) Xen HVM cc2.8xlarge Instance	Amazon Linux 2013.09 (EL6) Xen HVM c3.8xlarge Instance
CPUs	Intel(R) Xeon E5-2660 (2.2 GHz) 16 cores/node		Intel(R) Xeon E5-2670 (2.6 GHz) 16 cores/node		Intel(R) Xeon E5-2680v2 (2.8 GHz) 16 cores/node
RAM	64 GB DDR3			60.5 GB DDR3	
Interconnect	QDR4X InfiniBand Mellanox ConnectX-3		10GbE Mellanox ConnectX-3	10GbE (Xen Driver)	10GbE (Intel VF driver)

that governs how VFs share the underlying physical hardware on the I/O device is not prescribed by the SR-IOV standard, so hardware vendors can optimize coalescing of the VFs in both the hardware and the VF driver[7]. In Figure 1, this optimization logic is labeled "Sorter."

3. METHOD

3.1 Testing Environments

To evaluate the use of SR-IOV with InfiniBand for high-performance virtualization, we ran basic synthetic benchmarks on five test clusters summarized in Table 1.

In all cases, the nodes were interconnected with full-bisection bandwidth between them; this meant connecting all nodes on the same non-blocking switch for the native and InfiniBand tests and deploying all Amazon EC2 instances on a common placement group for the 10GbE tests. In addition, each virtual machine had access to the full resources of the underlying host hardware and did not have any contention from co-hosted virtual machines. However, no driver-level tuning was performed, and the results presented reflect the "out-of-box" performance. Possible performance improvements from such tunings are highlighted where applicable in the discussion.

3.2 Application Testing

To first understand the fundamental performance characteristics of the interconnects being evaluated, we used a subset of the OSU Micro-Benchmarks[22] (OMB version 3.9) compiled with OpenMPI 1.5 and GCC 4.4 to evaluate the latency, unidirectional bandwidth, and bidirectional bandwidth at the message passing layer. In addition, we used the HPCC ring tests[11] and OMB to simulate the effects of network congestion in select testing platforms.

Once a baseline for these fundamental communication modes was established, we chose to test two applications on several of the testing platforms: the Weather Research and Forecast (WRF) model[17] and Quantum ESPRESSO[5]. These applications represent opposite ends of the spectrum in terms of application communication profiles and provide upper and lower bounds of the overall performance impact of virtualization and SR-IOV on real application performance.

Both test applications were built with Intel Composer XE 2013 and all of the options necessary to allow the compiler to generate 256-byte AVX vector instructions that were available on all of the testing platforms. In addition, we used OpenMPI 1.5 for both InfiniBand and 10GbE platform tests, and we used Intel's Math Kernel Library (MKL) 11.0 to provide all BLAS, LAPACK, ScaLAPACK, and FFTW3 functions where necessary.

3.2.1 WRF - Weather Modeling

WRF is a widely used weather modeling application that is run in both research and operational forecasting. Its communication patterns are marked by an overwhelming amount of nearest-neighbor communications[15] where one MPI rank communicates with only a few other MPI ranks, and those communicating partners never change throughout the course of the simulation. Due to the way MPI ranks typically map to cores in most modern clusters, this means that much of the communication is actually done between MPI ranks that reside on the same physical host node, and the communication that does have to traverse the interconnect does so in highly predictable ways. Ultimately, this means that the interconnect experiences minimal congestion since communication is reliably patterned.

We used the CONUS-12km benchmark for actual performance evaluation[3] with WRF 3.4 over 96 cores (6 nodes). The size of the messages that WRF passes between the neighboring ranks for this benchmark are of intermediate size (predominantly between 4 KB and 32 KB), meaning they are neither critically dependent upon low latency or high-bandwidth interconnects. Thus, WRF and the CONUS-12km benchmark represent a "best-scale" problem in that its communication demands are modest, and the underlying algorithm is inherently scalable.

Finally, it is worth highlighting that although the WRF code is specific to weather modeling, the nearest-neighbor communications it performs are very similar to methods in many other domains such as molecular dynamics in materials science and finite-difference solvers in fluid dynamics.

3.2.2 Quantum ESPRESSO

Quantum ESPRESSO is an application that performs density functional theory (DFT) calculations for condensed matter problems. Unlike WRF, Quantum ESPRESSO and other DFT codes are inherently difficult to scale as a result of their algorithms having to perform extensive matrix diagonalization operations and 3D Fourier transforms.

Matrix diagonalization is done with the conjugate gradient algorithm which has very irregular communication patterns that put more pressure on the interconnect than WRF's nearest-neighbor communications. 3D Fourier transforms are similarly hard on the interconnect because they perform multiple matrix transpose operations where every MPI rank needs to send and receive all of its data to every other MPI rank. The very nature of these global collectives means they cause interconnect congestion, and efficient 3D FFTs are limited by the bisection bandwidth of the entire fabric connecting all of the compute nodes[14].

For this study, we chose to use the DEISA AUSURF112 benchmark[18] which has been extensively profiled in the past[20] with Quantum ESPRESSO 5.0.2 and 48 cores (3 nodes). This benchmark is dominated by `MPI_Alltoallv` and `MPI_Allreduce` calls, and the majority of message sizes are either large (> 64 KB) or small (< 16 bytes). The net result is that this test is critically dependent upon low-latency, high point-to-point bandwidth, and high total bisection bandwidth.

4. SR-IOV with 10GbE

4.1 Latency

As shown in Figure 2, SR-IOV provides a significant decrease in the latency of message passing over the 10GbE interconnect provided on Amazon EC2. For small messages (< 1K), the SR-IOV provided with Amazon's C3 instances results in 40% less latency which is a substantial improvement over the previous generation of cc2 instances that lacked SR-IOV. For larger messages, the bandwidth begins to dominate overall throughput and the relative improvement is not as great. However, even at the largest messages sizes tested (4 MB), SR-IOV delivered a minimum of 12% improvement in latency in these tests. These point-to-point results mirror the collective communication latency; both ring latency (random and ordered) and `MPI_Alltoallv` latency saw 40% improvement for small messages and between 12% and 16% improvement for large messages with SR-IOV.

Equally important as the average latency of message passing is the *variability* in I/O performance of VMs[4]. Because I/O devices that are not using SR-IOV or PCIe passthrough are interrupting

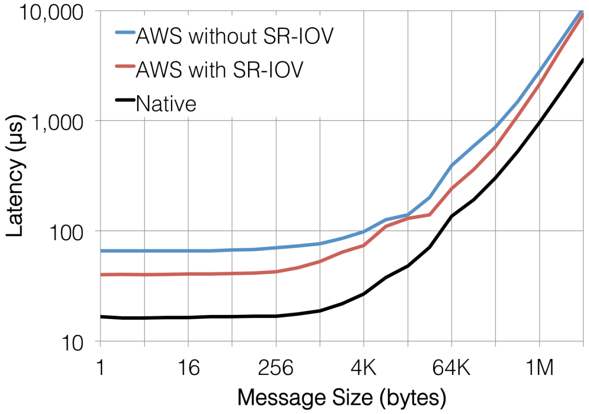

Figure 2. MPI point-to-point latency as measured by the `osu_latency` benchmark.

the CPU, there is a large amount of jitter that can occur as a result of background load on the host CPU. Figure 3 shows the latency measurements both with and without SR-IOV in the latency-bound range of messages sizes along with error bars bounding three standard deviations in the measurements made between all node-pairs in four-node clusters. For these smaller messages, SR-IOV provides 3× to 4× less variation in latency, but variation begins to increase for larger messages as variations in bandwidth also begin to affect the throughput.

Unfortunately, this 40% improvement in latency with SR-IOV still represents between 2× and 2.5× more latency than non-virtualized 10GbE (black line in Figure 2). While it is difficult to discern whether this large performance difference is simply a limitation of TCP over Ethernet via SR-IOV or an effect of Amazon EC2's cluster computing environment (we did not test SR-IOV with 10GbE on local hardware), the fact remains that the latency overhead experienced at the MPI layer on Amazon EC2 is significantly higher than using MPI on non-virtualized 10GbE. While there is some room for improvement in tuning the Linux kernel I/O stack that can potentially decrease SR-IOV latency, optimized tuning of the VMs was not tested in this study.

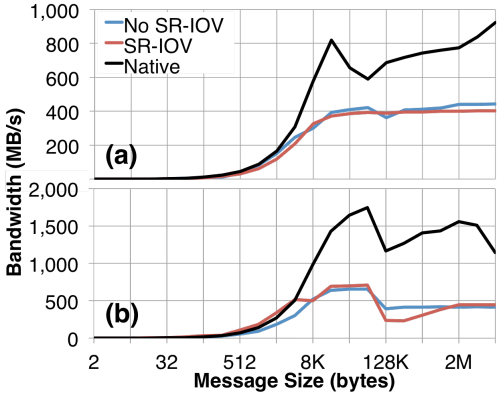

Figure 4. MPI (a) unidirectional bandwidth and (b) bidirectional bandwidth for QDR InfiniBand as measured by the `osu_bw` and `osu_bibw` benchmarks, respectively.

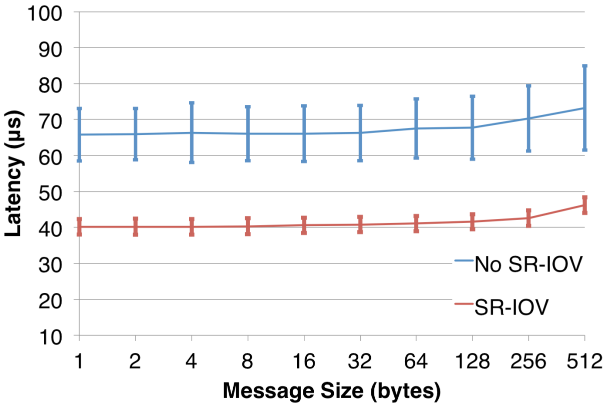

Figure 3. MPI point-to-point latency as measured by the `osu_latency` benchmark. Error bars are +/- three standard deviations from the mean.

4.2 Bandwidth

Although SR-IOV's primary benefit is to reduce the latency overhead of virtualized I/O, the latency data above also showed SR-IOV outperforming non-SR-IOV even for very large messages where bandwidth usually dominates. This suggests that 10GbE bandwidth on EC2 may also benefit from SR-IOV, but explicitly measuring MPI point-to-point bandwidth using `osu_bw` (Figure 4a) reveals that benefit of SR-IOV over non-SR-IOV virtualization is much less apparent when measuring bandwidth.

Messaging bandwidth never surpasses 500 MB/s, which is approximately 40% of line speed. By comparison, the non-virtualized 10GbE platform delivered between 1.5× and 2.0× the measurable bandwidth. The same holds for bidirectional bandwidth (Figure 4b); for intermediate-sized messages (M < 64 KB), data seems to flow at full duplex rates, but the bandwidth-limiting regime shows performance on par with unidirectional message passing. This is consistent with the notion that SR-IOV improves latency but not bandwidth, and it also reveals that the VMs in EC2 are very sensitive to bandwidth-heavy communication patterns.

Finally, we measured the collective bandwidth and found that SR-IOV afforded between 13% (random ring) and 17% (natural ring) higher bandwidth over Amazon's 10GbE, but the total ring bandwidth for 10GbE with SR-IOV on Amazon (330 MB/s for both ring tests) was less than half of the bandwidth available on the non-virtualized 10GbE test platform at SDSC (770 MB/s for both ring tests). In a practical sense, this means that algorithms such as 3D FFTs, which are bisection-bandwidth-limited, will perform poorly on EC2 regardless of if SR-IOV is being used. The Quantum ESPRESSO benchmark data presented in Section 6.2 support this.

Although we did not test the performance of SR-IOV with 10GbE in a dedicated testing environment independent of Amazon EC2, work by others[2,10] have found that 10GbE interfaces virtualized with SR-IOV are able to deliver over 90% of line rate, suggesting that these performance problems observed here are less telling of SR-IOV and more an effect of the Amazon EC2 environment.

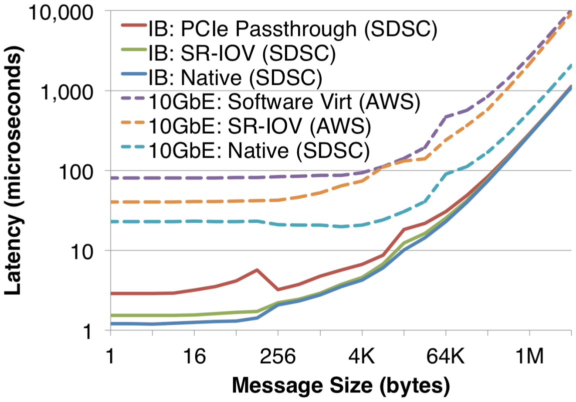

Figure 5. MPI point-to-point latency measured by osu_latency for QDR InfiniBand. Included for scale are the analogous 10GbE measurements from Amazon (AWS) and non-virtualized 10GbE.

5. SR-IOV with InfiniBand

5.1 Latency

Figure 5 depicts the latency measurements made over InfiniBand with and without SR-IOV. The latency difference is extremely small here—significantly less pronounced than the latency overhead of virtualized 10GbE, which is also included in the figure for scale. For very small messages (message size M < 128 bytes), the extra latency caused by virtualization is on the order of 30%. However, the extra latency overhead of virtualizing InfiniBand with SR-IOV is less than 10% for all messages larger than 128 bytes. Just as with SR-IOV and 10GbE, the overhead in latency diminishes to nearly zero once the fabric performance moves into the bandwidth-limited regime.

In addition to measuring the latency with SR-IOV, Figure 5 also includes latency measurements when virtualizing InfiniBand using the PF and PCIe passthrough. Although DMA via passthrough follows the same data path through the IOMMU to bypass the hypervisor, exposing the PF to the VM results in the native hardware driver being used instead of the paravirtualized VF driver. As detailed by Jose et al.[9], this exposes sub-optimal out-of-box performance to the VM and superior performance requires additional driver tuning.

However, the data in Figure 5 also clearly demonstrate that the performance of 10GbE inferior to QDR4X InfiniBand under all conditions; even virtualized InfiniBand has almost 10× less latency than non-virtualized 10GbE. In addition, although SR-IOV does incur some small increase in latency with InfiniBand, the performance loss is on the order of a few percent rather than orders of magnitude. While comparing 10GbE to InfiniBand is admittedly skewed by the lossy nature of TCP and Ethernet, this comparison does represent a reasonable comparison between HPC-centered technologies and the best-available commercial cloud alternative for HPC applications.

5.2 Bandwidth

As was found with 10GbE, SR-IOV does not affect the bandwidth available to VMs over InfiniBand. As shown in Figure 6, virtualization with SR-IOV incurs less than 2% loss of bandwidth across the entire range of message sizes. For the large messages, InfiniBand virtualized with SR-IOV is capable of delivering over

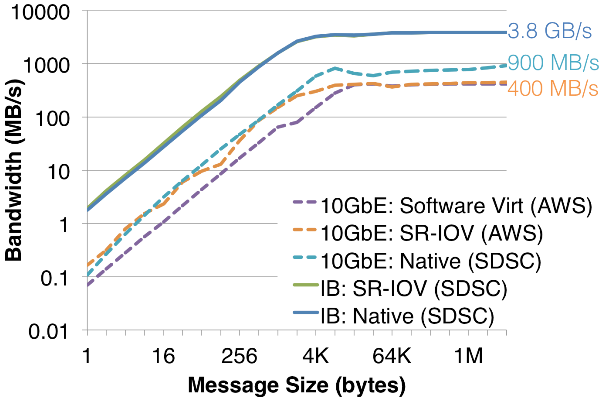

Figure 6. MPI point-to-point bandwidth measured by `osu_bw` for QDR InfiniBand. Included for scale are the analogous 10GbE measurements from Amazon (AWS) and non-virtualized 10GbE.

95% of the theoretical line speed. As labeled, InfiniBand over SR-IOV peaks out at 3.8 GB/s; by comparison, Amazon's 10GbE saturates at around 400 MB/s both with and without SR-IOV. This translates to performance differences of between 9x (large messages) and 25x (small messages) when message passing over virtualized InfiniBand instead of the virtualized 10GbE that exists in cloud-based compute services.

6. Application Benchmarks

Application testing was performed only on a subset of the test platforms given in Table 1. Namely, WRF and Quantum ESPRESSO were only run on unvirtualized InfiniBand, InfiniBand virtualized with SR-IOV, and software-virtualized 10GbE on AWS (cc2.8xlarge instances). This decision was motivated by three primary factors: (1) configuring c3.8xlarge clusters with SR-IOV on AWS was non-trivial due to the lack of Virtual Price Cloud (VPC) support in cluster provisioning frameworks such as StarCluster at the time of testing, (2) the SR-IOV-capable C3.8xlarge instances at the time of testing, resulting in the tests being relatively expensive, and (3) the faster Intel(R) Ivy Bridge-based processors used in c3.8xlarge instances have substantially different baseline performance than the Intel(R) Sandy Bridge-based processors in the InfiniBand platforms and the cc2.8xlarge (non-SR-IOV) instances.

6.1 WRF Performance

When the WRF CONUS-12km benchmark was run with six nodes (96 cores) over QDR4X InfiniBand virtualized with SR-IOV, the overall performance loss is only about 15% (Figure 7). Although not as low as the 10% overhead in MPI latency and roughly zero overhead in MPI bandwidth measured in the micro-benchmark tests of the previous section, a 15% increase in wall time is not an unacceptable loss if it allows users to develop an entire software ecosystem around using WRF (for example, with custom pre-built couplers) within a VM and simply deploy it as a scalable compute appliance on compute infrastructure supporting SR-IOV.

The performance loss when using software-virtualized 10GbE on AWS is also not excessive. With that being said though, both the IB (Native) and IB (SR-IOV) benchmarks were running on a test cluster with 2.2 GHz Intel(R) Xeon E5-2660 processors, while the Amazon instances were using 2.6 GHz Intel(R) Xeon E5-2670 processors. Thus, interconnect performance aside, the virtualized

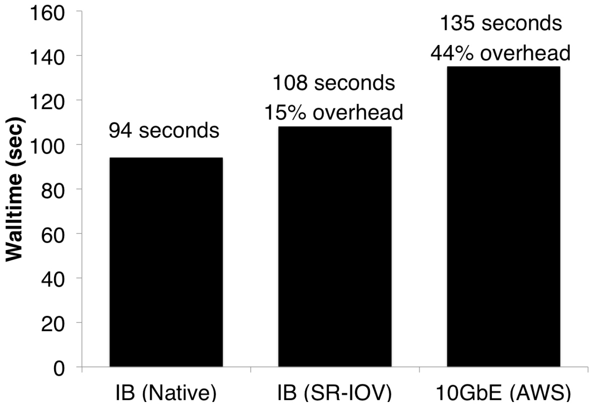

Figure 7. WRF CONUS-12km benchmark execution times across six 16-core nodes

10GbE performance did benefit from the fact that they were run on a platform with 18% higher clock frequency.

6.2 Quantum ESPRESSO Performance

Virtualizing the interconnect with SR-IOV contributes a substantially greater degree of overhead to the overall performance when calculating the DEISA AUSURF 112 benchmark with Quantum ESPRESSO. As shown in Figure 8, the execution time increases by 28% when run within a three-VM (48-core) virtual cluster despite the application being largely bandwidth-limited. This may be the effect of a large number of small messages (M < 16 bytes) that the benchmark generates, and it may also be a manifestation of the precipitous performance drop that SR-IOV appears to induce in collective operations reported by others[9].

While a 28% increase in execution time due to SR-IOV may border on unacceptable slowdowns, this benchmark yielded a slowdown of over 500% on Amazon's software-virtualized 10GbE. This is the result of Quantum ESPRESSO's extensive use of global and collective communication which are limited by bisection bandwidth and congestion sensitivity. As discussed in Section 4.2, the deliverable collective bandwidth on both VMs with and without SR-IOV enabled for their 10GbE interfaces on EC2 remains low; thus, we do not expect the Quantum ESPRESSO performance to be substantially better on c3.8xlarge instances with SR-IOV enabled.

7. Conclusions

The use of SR-IOV to virtualize the 10GbE connections between VMs does have significant, measurable performance improvements software-virtualized I/O in terms of latency on AWS. The jitter in the latency is also substantially reduced, and SR-IOV does represent a big step forward in increasing the performance of EC2's cluster compute capabilities. However, these instances with SR-IOV still display over twice the latency of non-virtualized 10GbE and minimal improvement in the bandwidth available over software-virtualized interfaces. Cluster compute instances are only able to achieve around 40% of the theoretical peak bandwidth even with SR-IOV, and congestion caused by full-duplex communication and all-to-all message passing severely degrades the overall I/O performance.

The use of SR-IOV to virtualize InfiniBand has dramatic performance benefits though. Overhead in latency caused by SR-IOV remains below 10% for all but the smallest of messages, and MPI bandwidth overhead is negligible (< 2%) and allows VMs to achieve > 95% line speed.

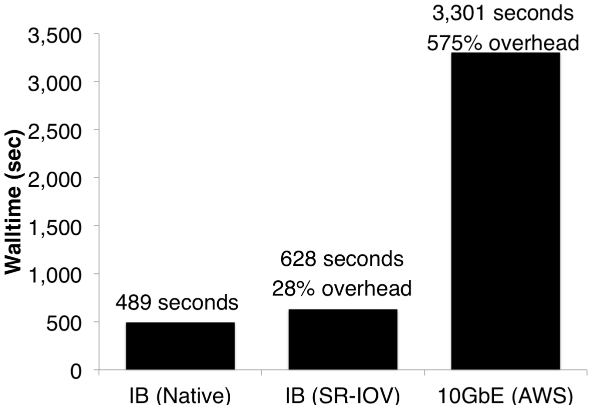

Figure 8. Quantum ESPRESSO DEISA AUSURF112 benchmark execution times across three 16-core nodes

SR-IOV ultimately improves the latency of message pasing, and applications that are bound by latency rather than bandwidth will benefit the most. With InfiniBand, the application performance overhead caused by SR-IOV will typically be on the order of 10% (for nearest-neighbor type communication like with WRF) to 50% (for global/collective type communication as with Quantum ESPRESSO) for smaller problems ranging from three to eight nodes.

This represents a major performance advantage over software-virtualized VMs found in commercial clouds. Using SR-IOV with InfiniBand delivers raw performance that is orders-of-magnitude better in both latency and bandwidth than commercial cloud offerings. SR-IOV represents a major step forward in the pursuit of high-performance virtualization within high-performance computing and makes the flexibility offered by deploying virtualized compute appliances tractable for multi-node parallel applications.

8. ACKNOWLEDGMENTS

The authors wish to acknowledge operational support from Christopher Irving. meaningful discussion and technological guidance from Philip Papadopoulos, and editorial contributions from Robert Sinkovits.

9. REFERENCES

1. Bollen, J., Fox, G., and Singhal, P.R. How and where the TeraGrid supercomputing infrastructure benefits science. *Journal of Informetrics 5*, 1 (2011), 114–121.

2. Dong, Y., Yang, X., Li, J., Liao, G., Tian, K., and Guan, H. High performance network virtualization with SR-IOV. *Journal of Parallel and Distributed Computing 72*, 11 (2012), 1471–1480.

3. Eldred, C. (Pittsburgh S.C. and Michalakes, J. (National C. for A.R. WRF V3 Parallel Benchmark Page. 2008. http://www.mmm.ucar.edu/wrf/WG2/benchv3/.

4. Ghoshal, D., Canon, R.S., and Ramakrishnan, L. I/O performance of virtualized cloud environments. *Proceedings of the Second International Workshop on Data Intensive Computing in the Clouds*, ACM Press (2011), 71–80.

5. Giannozzi, P., Baroni, S., Bonini, N., et al. QUANTUM ESPRESSO: a modular and open-source software project for quantum simulations of materials. *Journal of Physics: Condensed Matter 21*, 39 (2009), 395502.

6. Hines, M.L. and Carnevale, N.T. Expanding NEURON's Repertoire of Mechanisms with NMODL. *Neural Computation 12*, 5 (2000), 995–1007.

7. Huang, Z., Ma, R., Li, J., Chang, Z., and Guan, H. Adaptive and Scalable Optimizations for High Performance SR-IOV. *2012 IEEE International Conference on Cluster Computing*, IEEE (2012), 459–467.

8. Jackson, K.R., Ramakrishnan, L., Muriki, K., et al. Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, IEEE (2010), 159–168.

9. Jose, J., Li, M., Lu, X., Kandalla, K.C., Arnold, M.D., and Panda, D.K. SR-IOV Support for Virtualization on InfiniBand Clusters: Early Experience. *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, IEEE (2013), 385–392.

10. Liu, J. Evaluating standard-based self-virtualizing devices: A performance study on 10 GbE NICs with SR-IOV support. *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, IEEE (2010), 1–12.

11. Luszczek, P.R., Bailey, D.H., Dongarra, J.J., et al. The HPC Challenge (HPCC) benchmark suite. *Proceedings of the 2006 ACM/IEEE conference on Supercomputing - SC '06*, ACM Press (2006), 213.

12. Mehrotra, P., Djomehri, J., Heistand, S., et al. Performance evaluation of Amazon EC2 for NASA HPC applications. *Proceedings of the 3rd workshop on Scientific Cloud Computing Date - ScienceCloud '12*, ACM Press (2012), 41–49.

13. Miller, M.A., Pfeiffer, W., and Schwartz, T. Creating the CIPRES Science Gateway for inference of large phylogenetic trees. *Gateway Computing Environments Workshop (GCE), 2010*, (2010).

14. Pekurovsky, D. P3DFFT: A Framework for Parallel Computations of Fourier Transforms in Three Dimensions. *SIAM Journal on Scientific Computing 34*, 4 (2012), C192–C209.

15. Shainer, G., Liu, T., Michalakes, J., et al. Weather Research and Forecast (WRF) Model: Performance Analysis on Advanced Multi-core HPC Clusters. *The 10th LCI International Conference on High-Performance Clustered Computing*, (2009).

16. Sivagnanam, S., Astakhov, V., Yoshimoto, K., et al. A neuroscience gateway: software and implementation. *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment Gateway to Discovery - XSEDE '13*, ACM Press (2013).

17. Skamarock, W.C., Klemp, J.B., Dudhia, J., et al. *NCAR Technical Note NCAR/TN–475+STR: A Description of the Advanced Research WRF Version 3*. Boulder, CO, 2008.

18. QuantumESPRESSO - Deisa. 2008. http://www.deisa.eu/science/benchmarking/codes/quantumespresso.

19. Single Root I/O Virtualization and Sharing 1.1 specification. 2010. http://www.pcisig.com/specifications/iov/single_root/.

20. HPC Advisory Council Best Practices. 2010. http://www.hpcadvisorycouncil.com/best_practices.php.

21. AMD I/O Virtualization Technology (IOMMU) Specification. 2011. http://developer.amd.com/wordpress/media/2012/10/48882.pdf.

22. OSU Microbenchmarks. 2011. http://mvapich.cse.ohio-state.edu/benchmarks/.

23. Intel Virtualization Technology for Directed I/O. 2013. http://www.intel.com/content/dam/www/public/us/en/documents/product-specifications/vt-directed-io-spec.pdf.