Performance of RDMA and HPC Applications in Virtual Machines using FDR InfiniBand on VMware vSphere®

TECHNICAL WHITE PAPER

Table of Contents

Introduction

High performance computing (HPC) helps scientists and engineers solve complex problems with powerful compute resources and high speed interconnects. In recent years, there has been a growing interest in virtualizing HPC environments, particularly due to its value for improving scientific productivity, through capabilities such as data and resource isolation, efficient infrastructure utilization, effective resource management, greater flexibility and good performance [1].

This paper, as an update to the previously published performance white paper "RDMA Performance in Virtual Machines using QDR InfiniBand on VMware vSphere 5", evaluates latencies of remote direct memory access (RDMA) and the Message Passing Interface (MPI) library using Mellanox FDR InfiniBand adaptors in passthrough mode with vSphere. Delivering low-latency, high performance solutions in cloud environments has been thought an insurmountable barrier for most of cloud solutions. In our tests, best performance in virtual machines is very close to native, whether at the RDMA or MPI level. Based on RDMA and MPI performance, selected HPC applications are tested and performance improvements are verified. The results demonstrate significant performance improvements over a range of VMware ESXi hypervisor versions.

Before discussing the performance tests and results, let's briefly review some background regarding virtualizing HPC.

Cloud Model for HPC

Virtualized server resources can be consumed either on-premise or off-premise. While off-premise approaches – public or private cloud – can be a useful approach for organizations not in a position to create and manage their own on-premise HPC environment, for those with existing HPC environments and expertise these venues are less attractive. While these clouds can sometimes be used to temporarily augment on-premise resources, replacing on-premise HPC resources with off-premise capabilities is often not a viable approach for several reasons. First, the economics of renting equivalent replacement resources can be unattractive. Second, the data volumes related to many HPC workloads and environments are such that data transfers between sites is not a tenable approach, and, third, data security and compliance can be issues.

On-premise virtualized environments – with or without a private cloud management layer – offer the ability to tune the underlying virtualization platform to meet an organization's workload performance requirements in ways that are not exposed in off-premise rental environments. Thus, excellent performance can be achieved while simultaneously addressing security and data volume issues and delivering a more flexible and agile solution to an organization's end-users.

High Throughput versus Parallel Distributed Applications

There are two main categories of HPC workloads: high-throughput oriented or parallel distributed applications. Throughput-oriented workloads are usually categorized as "embarrassingly parallel" and require no communication or synchronization between jobs. Typical throughput applications include Monte Carlo simulations, bioinformatics and other parameter-variation simulations, where the same program has hundreds or thousands even millions of executions with varying inputs. With advances in both vSphere as well as x86 microprocessor architecture, throughput applications can generally run at close to full speed in the VMware virtualization environment – with less than 5% performance degradation compared to native, often just 1~2%. This has been verified through tests of various applications across multiple disciplines, including life sciences, electronic design automation, finance, etc.

Parallel distributed applications, in contrast, often perform sustained and intense communication between job components, making their performance sensitive to interconnect bandwidth and latency.

RDMA and MPI

Low latency is a critical characteristic of interconnects used for distributed systems. Remote Direct Memory Access (RDMA) allows a network adapter to transfer data directly to or from application memory without involving the operating system, thus enabling high bandwidth and low latency. This design makes it a popular option for HPC system interconnects. RDMA is also used for distributed database implementations as well as other Enterprise use-cases in which low latency and high bandwidth are required to achieve good performance across multiple nodes. Implementations of RDMA include InfiniBand, RDMA over Converged Ethernet (RoCE) and iWARP.

Message Passing Interface (MPI) was designed to enable parallel programming using a message-passing paradigm on distributed-memory systems. It has become a standard for multiple-processor programming of HPC codes that runs on a variety of machines – from small clusters to giant supercomputers. Numerous scientific applications that run in a distributed way use the MPI library and take advantage of communication primitives such as point-to-point operations (e.g., *MPI_Send* and *MPI_Receive*) and collective operations (e.g., *MPI_Bcast*, *MPI_Reduce*). MPI libraries are designed to use the best available pathways between communicating endpoints, including shared memory, Ethernet, and – for high performance – RDMA-capable interconnects.

VMware DirectPath I/O

DirectPath I/O (initially released in vSphere 4.0) allows PCI devices, such as physical network cards, to be made directly visible to guests. Single Root I/O Virtualization (SR-IOV, initially released in vSphere 5.1) allows multiple VMs to share single PCI device, without sacrificing runtime performance. This is beneficial for workloads that require very low latency or high packet rates. It should be noted that while they can boost virtual performance, neither DirectPath I/O or SR-IOV are compatible with certain VMware platform features such as memory overcommit, vMotion, snapshots, and Network I/O Control.

Performance Tests

RDMA

We performed a series of point-to-point tests using two hosts connected through a Mellanox 12-port InfiniBand/Ethernet Switch. Each host was equipped with a Mellanox ConnectX-3 FDR (56 Gb/s) InfiniBand adaptor, which also supports 40 Gbs RoCE. Native-to-native tests were run using Red Hat Enterprise Linux on each host, while the virtual-to-virtual tests were run with an ESXi hypervisor running on each host along with a single virtual machine running the same Red Hat version. The InfiniBand devices were configured in ESXi as passthrough devices using VMDirectPath I/O [2]. All performance results presented here represent the best virtual-to-virtual performance we have measured using FDR InfiniBand in passthrough mode. To achieve optimal performance, best practices in terms of BIOS and ESXi hosts settings have been observed. Full details on hardware and software configurations and vSphere special settings are given in the Configuration Details section.

We installed the Mellanox OpenFabrics Enterprise Distribution for Linux (MLNX_OFED) [3] and ran RDMA tests using its micro-benchmark suite, including latency tests *read_lat*, *send_lat* and *write_lat.* All tests were run in Reliable Connected (RC) mode and using polling based I/O completions. Each reported data point represents the mean of 10000 iterations at that message size and half round-trip numbers are reported.

Point-to-Point MPI

We performed MPI communication performance tests using the MVAPICH library [4], whose development is lead by Ohio State University (OSU) for the purpose of providing optimized RDMA communication "over InfiniBand, 10GibE/iWARP and RDMA over Converged Ethernet."

MPI-level latencies were tested using *osu_latency*, a part of the OSU micro-benchmark suite. *osu_latency* measures a range of message sizes (*2B ~ 8MB*) in a ping-pong fashion using blocking version of MPI functions (*MPI_Send* and *MPI_Recv*). Each data point represents the mean of 10000 iterations at that message size and half round-trip number is reported.

MPI Applications

Based on the above point-to-point RDMA and MPI tests, we ran a set of MPI applications to assess performance at a higher level. Our testbed is an eight-node cluster with 16 cores per node. Using a strong-scaling methodology, we solved problems of a fixed size using from eight MPI processes (1 per node) to 128 MPI processes (16 per node).

In this report, we present the performance of two benchmarks: Weather Research and Forecasting (WRF) and Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS). WRF is a popular numerical solver designed for atmospheric research and LAMMPS is a classical molecular dynamics code that is used for a variety of material simulations. We focus on these two particular benchmarks because they have intensive communication patterns (communication time is larger than 10% of total simulation time for a typical simulation) and thus we can study the performance affects of virtualization on low-latency communication in real HPC scenarios.

Performance Results

RDMA

Figure 1 and Figure 2 show the RDMA read and send latency comparisons between Native (un-virtualized), ESXi 6.0, ESXi 6.0u1, and EngBuild. ESXi 6.0 and 6.0u1 are currently-shipping versions of the VMware hypervisor. EngBuild is an internal ESX engineering build. RDMA write latency results are generally similar to RDMA send performance and, thus, are not shown. The figures focus on small messages since latency is most critical factor in this range of message sizes.

From the figures we can see:

- For RDMA read latency, ESXi 6.0 has ~0.8 μs overhead relative to native. But this overhead has been eliminated in ESXi 6.0u1.
- RDMA send latency shows a ~0.4 μs overhead, and there is also an anomalous latency increase in the range of message sizes 16 to 128 bytes in ESXi 6.0. However, both of these issues have been addressed as of ESXi 6.0u1.

The latency improvements shown in ESXi 6.0u1 and EngBuild are achieved by implementing support for writing combining (WC), which takes advantage of a hardware feature of Intel processors[1]. With write combining support implemented, we are able to achieve close-to-native latencies for all message sizes using VMware Direct Path I/O. In other testing, we have shown that writing combining support also improves InfiniBand SR-IOV latency in a similar way. Since Mellanox does not have official support for FDR InfiniBand SR-IOV on ESXi, we will skip this discussion in this report. In future, we will evaluate SR-IOV performance with EDR InfiniBand.

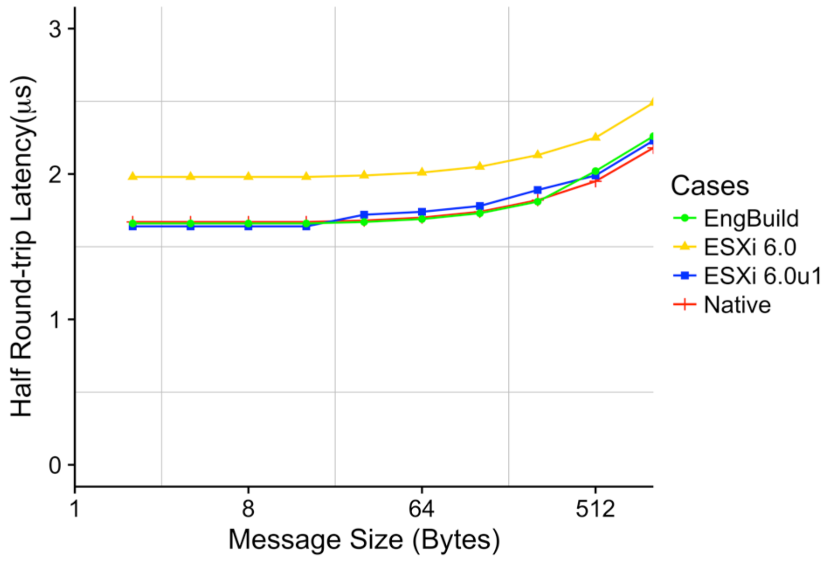

Figure 1: InfiniBand RDMA read latencies for Native and three VMware ESXi hypervisor versions. Lower is better.

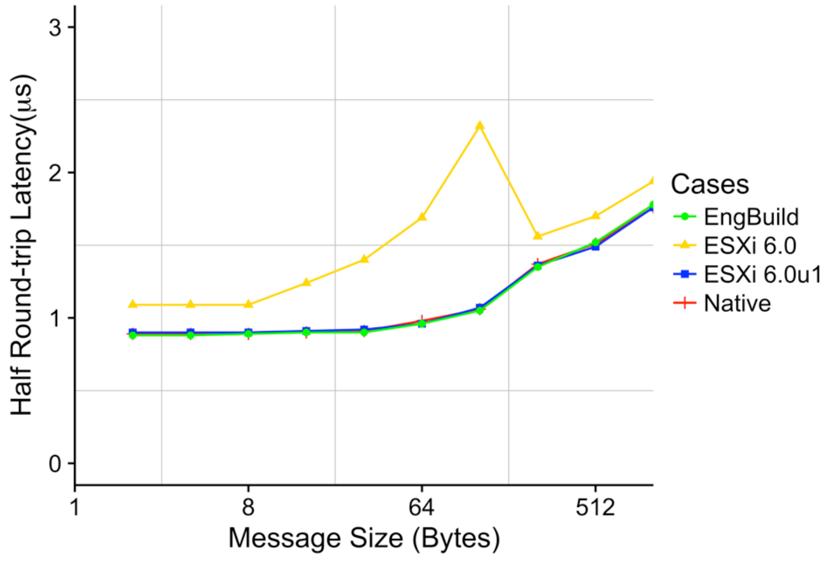

Figure 2: InfiniBand RDMA send latencies for Native and three VMware ESXi hypervisor versions. Lower is better.

1 AMD processors also support write combining, but current ESX support for this feature is available for Intel processors only.

Point-to-Point MPI

Figure 3 shows the MPI small-messages latency comparisons between Native, ESXi 6.0, ESXi 6.0u1, and EngBuild.

From the figure we can see:

- With the support of writing combining that has been incorporated in ESXi 6.0u1, a large portion of the latency overhead seen with ESXi 6.0 has been removed. However, there is still ~20% latency overhead relative to native at the MPI level.
- This 20% overhead has been eliminated in EngBuild.

The MPI latency performance improvements from ESXi 6.0u1 to EngBuild have been achieved by implementing support for Large Intel VT-d Pages. Specifically, EngBuild includes support for 2MB pages in Intel VT-d page tables. Because the relationship between large-page support and MPI small-message latency overheads is not obvious, we describe the link below.

The latency benchmark measures latency over a range of message sizes from two bytes to eight megabytes, iterating 10000 times for each message size to generate averaged results. The benchmark uses a single, large, pre-allocated buffer for all message transfers. However, MPI implementations, including MVAPICH, OpenMPI and Intel MPI, use separate protocols for handling small messages and large messages over RDMA interconnects. While it is true that RDMA semantics allow any user buffer to be registered as a transfer buffer to avoid buffer copying, in practice it is expensive to register a memory region. Because of this expense, MPI libraries transfer small messages by creating a set of pre-registered small transfer buffers and copying messages into those transfer buffers in a round-robin fashion rather than registering a user-space buffer for RDMA. For small messages, the extra cost of transferring the data from a user buffer to a transfer buffer is still less than the cost of buffer registration. This cycling through a large set of pre-registered buffers will lead to IOTLB misses; the support for large pages reduces the miss rate, resulting in lower latency.

Overall, with the progression of ESXi hypervisor versions, we are able to achieve close-to-native MPI communication performance for all message sizes in Direct Path I/O (passthrough) mode. We expect to see performance improvements for parallel-distributed HPC applications as well, since most of these rely on low MPI latencies to varying extents.

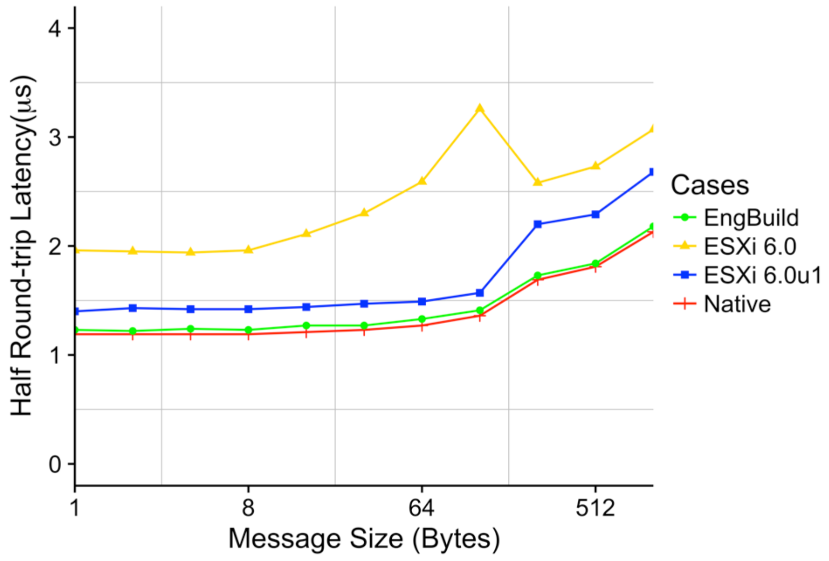

Figure 3: MPI small messages latencies for Native and three VMware ESXi hypervisor versions. Lower is better.

MPI Applications

Figure 4 and Figure 5 show performance ratios for WRF and LAMMPS running with ESXi 6.0 and EngBuild versus Native, respectively.

These results reveal that:

- The maximum degradation is less than 5% for 8 to 64 MPI processes for both applications.
- At higher scale, running with 128 MPI processes leads to more performance degradation due to the increased communication involved.
- With support for both write combining and Intel Large VT-d Pages, EngBuild demonstrates better performance than ESXi 6.0, especially for larger numbers of MPI processes.

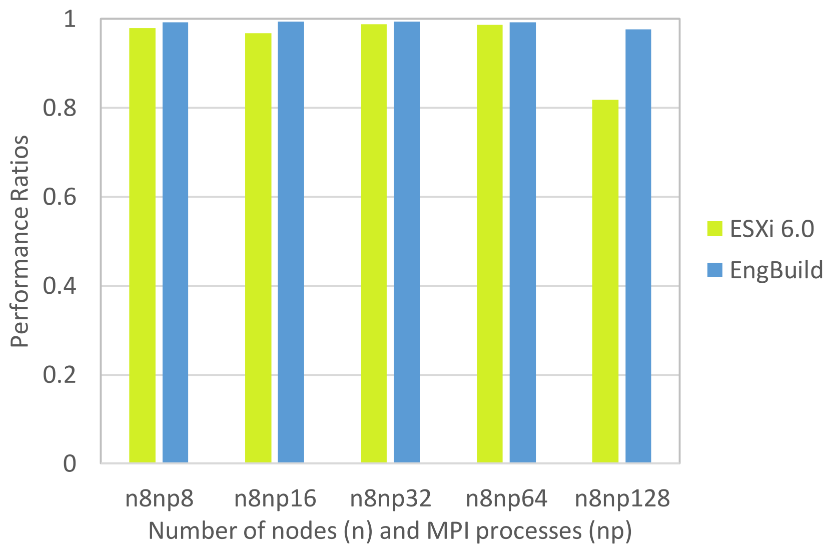

Figure 4: Performance ratios of WRF using ESXi 6.0 and EngBuild versus Native. Higher is better, with 1.0 indicating no performance loss for virtualization.

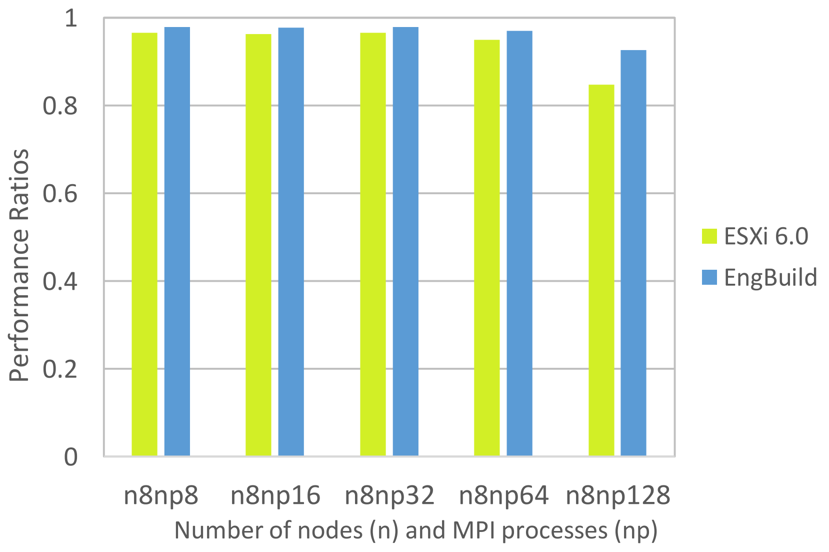

Figure 5: Performance ratios of LAMMPS using ESXi 6.0 and EngBuild versus Native. Higher is better, with 1.0 indicating no performance loss for virtualization.

Conclusion

Due to both engineering improvements at VMware and continued hardware innovation, we have been able to eliminate RDMA and MPI latency overheads for the smallest InfiniBand RDMA and MPI latencies of interest to customers running very latency-sensitive distributed applications.

For HPC, low latency is fundamental. Having achieved low virtual latencies, we have shown that MPI applications of modest scale can be run very well in a VMware virtualized environment. As VMware continues to work to optimize performance, we expect the performance of a wider range and higher scale of HPC applications to improve, further expanding the applicability of virtualization to High Performance Computing.

Configuration Details

Hardware

- HP ProLiant DL380 Gen8
- Two-socket Intel E5-2667v2 (3.0GHz/8core/25MB/130W) and 128 GB Memory
- 3 HP 600GB 6G SAS 10K rpm SFF (2.5-inch)
- Mellanox ConnectX-3 FDR InfiniBand/RoCE Adapter
- Mellanox 12-port InfiniBand/Ethernet Switch

BIOS Settings

- HP Power Profile: "Maximum Performance"
- HP Power Regulator: "Static High Performance Mode"
- Redundant Power Supply: "Balanced Mode"
- Minimum Processor Idle Power Core State: "No C-states"
- Minimum Processor Idle Power Package State: "No Package State"
- Memory Power Saving Mode: “Maximum Performance"
- Processor Hyper-Threading: “Disabled”
- Intel VT-d: “Enabled”

Software

- Guest OS is 64-bit RHEL 2.6.32-431.17.1.el6.x86_64 for both native and virtual
- Mellanox OpenFabrics Enterprise Distribution MLNX_OFED_LINUX-2.2-1.0.1 is installed on every physical and virtual node to provide the InfiniBand interface
- Hypervisors are ESXi 6.0 (build number 1921427), ESXi 6.0 update1(build number 2781041), EngBuild (an internal engineering build)
- OSU benchmark version 4.3
- MVAPICH library version 2-2.0rc1

vSphere Settings

- In a multi-socket NUMA architecture system, lowest latencies are achieved by accessing local, socket-attached memory and by using the socket to which required PCI devices are attached (this non-uniformity of I/O is a feature of newer Intel processors). Thus, in order to achieve optimal performance, processor affinity for vCPUs was used to ensure the test VM was scheduled on the NUMA node to which the InfiniBand device is attached, and that the VM’s memory is allocated from the NUMA node’s local memory. This can be achieved using the vSphere client tab “VM Options” >tab “Advanced”-> tab “Edit Configuration” and adding entries for "numa.nodeAffinity=0" where 0 is the processor socket number to which the InfiniBand device is attached [5].

- In our point-to-point tests, each VM was configured with 8vCPUs and 58GB in order to fit into one processor socket.

References

[1] J. Simons, E. DeMattia, and C. Chaubal, "Virtualizing HPC and Technical Computing with VMware vSphere," *VMware Technical White Paper*, http://www.vmware.com/files/pdf/techpaper/vmware-virtualizing-hpc-technical-computing-with-vsphere.pdf.

[2] "Configuration Examples and Troubleshooting for VMDirectPath," *VMware Technical Note* http://www.vmware.com/pdf/vsp_4_vmdirectpath_host.pdf.

[3] "Mellanox OFED for Linux User Manual," http://www.mellanox.com/related-docs/prod_software/Mellanox_OFED_Linux_User_Manual_v2.2-1.0.1.pdf.

[4] "MVAPICH2 2.1 User Guide," http://mvapich.cse.ohio-state.edu/static/media/mvapich/mvapich2-2.1a-userguide.pdf.

[5] B. Davda, "Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs," *VMware Technical White Paper*, http://www.vmware.com/files/pdf/techpaper/VMW-Tuning-Latency-Sensitive-Workloads.pdf.

Contributors

Na Zhang is member of technical staff working on High Performance Computing within VMware's Office of the CTO. She received her Ph.D. degree in Applied Mathematics from Stony Brook University. Her research primarily focused on design and analysis of parallel algorithms for large- and multi- scale simulations running on supercomputers. She previously had two summer internships working on HPC at VMware and performed various of benchmark tests. This work was part of her 2015 internship project.

Josh Simons currently leads an effort within VMware's Office of the CTO to bring the full value of virtualization to HPC. He has over 20 years of experience in High Performance Computing. Previously, he was a Distinguished Engineer at Sun Microsystems with broad responsibilities for HPC direction and strategy. He joined Sun in 1996 from Thinking Machines Corporation, a pioneering company in the area of Massively Parallel Processors (MPPs), where he held a variety of technical positions. Josh has worked on developer tools for distributed parallel computing, including language and compiler design, scalable parallel debugger design and development, and MPI. He has also worked in the areas of 3D graphics, image processing, and real-time device control. Josh has an undergraduate degree in Engineering from Harvard College and a Masters in Computer Science from Harvard University. He has served as a member of the OpenMP ARB Board of Directors since 2002.

Acknowledgements

We are thankful to Bhavesh Davda and Doug Covelli for their suggestions and discussions on this performance project, to Dave Dunn for development work, and Bryan Connery for his technical support.