DOI: 10.14529/jsfi150303

Performance Assessment of InfiniBand HPC Cloud Instances on Intel® Haswell and Intel® Sandy Bridge Architectures

Jonathan Low[1]***, Jakub Chrzeszczyk***[2]***, Andrew Howard***[2]***, Andrzej Chrzeszczyk***[3]

This paper aims to establish a performance baseline of an HPC installation of OpenStack. We created InfiniCloud - a distributed High Performance Cloud hosted on remote nodes of InfiniCortex. InfiniCloud compute nodes use high performance Intel (R) Haswell and Sandy Bridge CPUs, SSD storage and 64-256GB RAM. All computational resources are connected by high performance IB interconnects and are capable of trans-continental IB communication using Obsidian Longbow range extenders.

We benchmark the performance of our test-beds using micro-benchmarks for TCP bandwidth, IB bandwidth and latency, file creation performance, MPI collectives and Linpack. This paper compares different CPU generations across virtual and bare-metal environments.

The results show modest improvements in TCP and IB bandwidth and latency on Haswell; performance being largely dependent on the IB hardware. Virtual overheads were minimal and near-native performance is possible for sufficiently large messages. From the Linpack testing, users can expect the performance in their applications on Haswell-provisioned VMs more than twice. On Haswell hardware, native and virtual performance differences is still significant for MPI collective operations. Finally, our parallel filesystem testing revealed virtual performance coming close to native only for non-sync/fsync file operations.

Keywords: Cloud-Computing, InfiniBand, Trans-continental, Benchmarking, Virtualization, SRIOV, BeeGFS, OpenStack, HPC.

Introduction

Cloud computing offers resources on-demand as an Infrastructure-as-a-Service (IaaS) platform, providing good flexibility in resource allocation and usage that can be easily managed by both end-users and administrators. This brings the benefits of isolated, user-customised software and hardware environments that enable software reproducibility and turn-key solutions and applications regardless of the underlying physical computing hardware. Over the past few years, there has been a shift in utilising such cloud computing services and its associated benefits to address the needs of the HPC scientific community [7].

Since 2009, the National Computational Infrastructure (NCI) in Australia have been providing a cloud computing platform service for compute and I/O-intensive workloads to their big data research community [2]. NCI Cloud services provide computational resources in the form of virtual machines (VM) provisioned by the OpenStack[4] cloud operating system platform. The bare-metal (BM) backend consists of high-spec Intel CPUs, SSDs for storage and started off with 10Gb Ethernet for networking. Encouraged by rapid adoption of the Cloud services, NCI enhanced the interconnect from 10Gb to 56Gb Ethernet using Mellanox hardware together with Single Root IO Virtualisation (SR-IOV) as a first phase. This brings significant performance improvements to traditional HPC applications that typically require a fast interconnect. As the same Mellanox hardware was capable of 56Gb InfiniBand (IB) and SR-IOV, A*CRC and NCI

[1]A*STAR Computational Resource Centre, Singapore
[2]National Computational Infrastructure, ANU, Canberra, Australia
[3]Jan Kochanowski University, Kielce, Poland
[4]OpenStack cloud computing platform - http://www.openstack.org

teams worked together to build InfiniCloud, a native IB OpenStack Cloud prototype which was completed in October 2014 and demonstrated at SC14 in New Orleans, as a part of the InfiniCortex project [11]. In February 2015 A*CRC and NCI teams further enhanced InfiniCloud by addition of six SGI servers located in Singapore. They consist of the latest Intel Haswell CPU models, DDR4 memory, SSD storage and the same Mellanox 56Gb ConnectX-3 hardware. The servers at both NCI and A*CRC can communicate with each other with native InfiniBand through range-extender equipment from Obsidian Strategics. InfiniCloud users can quickly and easily make use of compute resources located at either location, despite the bare-metal hardware residing on a remote site. This allows to distribute the processing of user data, as well as utilizing additional capacity and unique capabilities of hardware located at each site. For example, users can opt for top-performance CPUs in Singapore or larger available memory fat-nodes in Australia.

In this paper we aim to provide an insight into the improved performance that users can expect when moving from the NCI SandyBridge hardware to A*CRC's Haswell servers. Thus we present bandwidth, latency and MPI micro-benchmarks to gauge the VM network performance, storage benchmarks to test the VM storage backend and Linpack benchmarks to gauge real HPC application performance.

The paper is presented as follows. Section 1 explores any past work done and how this paper's work relates to that. Section 2 explains the hardware and software configuration as well as details of the benchmarks performed. Section 3 presents the results obtained, followed by concluding remarks in the last section.

1. Background and Related Work

In the past, virtualised environments incurred significant overheads so that their use for intensive workloads came with significant performance degradation. This started to improve, starting with the introduction of Intel VT to better share resources and improving the performance of CPU, memory virtualisation and more. However, network I/O remained a challenge to obtain near-native performance amongst virtual machines due to the packet processing, switching and CPU interruptions involved. These overheads become very significant when attempting to make use of high speed interconnects that typical HPC workloads require and their associated features such as RDMA that needed to work effectively in virtual environments.

To solve the network I/O problem, the SR-IOV technology was drawn up by the PCI Special Interest Group. This is the hardware-based virtualisation method that allows near-native performance of network interfaces to be realised, where network I/O can bypass the hypervisor to avoid involvement of the CPU. This works for both Ethernet and InfiniBand. Amazon Web Services provide SRIOV-enabled Gigabit Ethernet (GigE) for their C3 instances[5], the feature marketed as "Enhanced Networking" and there have been numerous performance studies for SRIOV-enabled Gigabit Ethernet and InfiniBand usage [3, 6, 8–10].

Today there exists a number of virtual environment installations utilising InfiniBand, ours included. Citing other examples:

- A private cloud platform, "FermiCloud", was used to study SRIOV-enabled, IB-interconnected virtual hosts provisioned using OpenNebula and conducting MPI micro-

[5]Announcing New Amazon EC2 Compute Optimized Instances - http://aws.amazon.com/about-aws/whats-new/2013/11/14/announcing-new-amazon-ec2-compute-optimized-instances

benchmarks and HPL [3]. The hardware used in the study were Intel Westmere CPUs and DDR InfiniBand hardware.

- An in-depth performance study on SRIOV-enabled FDR InfiniBand for virtual clusters examined the behaviour of virtual IB under differing combinations of resource subscriptions, IB progression modes and parallel programming languages [8].
- The San Diego Supercomputing Center will host a Pflop-capable HPC resource with a key aim to

> *"Provide a virtualized environment to support development of customized software stacks, virtual environments, and project control of workspaces"* [10]

For our exercise, we use the OpenStack cloud operating system to provision resources and test the performance of a SRIOV-enabled InfiniBand virtual cluster on SandyBridge & Haswell CPUs with FDR InfiniBand.

2. Setup

In this section we detail the hardware and software setup and provide details of the benchmarking configuration.

2.1. Setup of NCI SandyBridge and A*CRC Haswell VMs

The hardware details of both bare-metal server types are summarised in tab. 1.

Table 1. Summary of NCI and A*CRC server specifications, provisioned and managed by the OpenStack Icehouse release

	OpenStack (IceHouse) provisioned	
Location	NCI, Australia	A*CRC, Singapore
CPU	2x Intel E5-2650 8-core SandyBridge (SB) Arch.	2x Intel E5-2680v3 12-core Haswell (HW) Arch.
Memory	256GB 1333MHz DDR3	128GB 2133MHz DDR4
Storage	6x 10k RPM Seagate HDD	Intel DC S3500 SSD
Network	Mellanox Connect-X3 FDR	Mellanox Connect-X3 FDR
Operating System	CentOS 6.5	CentOS 6.5
# of compute servers used	4 (2 + 2: BM-BM and VM-VM)	4 (2 + 2: BM-BM and VM-VM)

To compare native and virtual performance, the four servers were used as two pairs, one for BM testing whilst the other was used with one VM instance each. Mellanox OFED[6] drivers v2.4 were used to provide the hardware-based SR-IOV virtualisation of the InfiniBand interface in the form of virtual functions that can be dedicated to particular VM instances. As of March 2015, SR-IOV is a requirement for running InfiniBand in virtual instances on OpenStack. In standalone KVM, non-OpenStack virtualized environments, it is possible to assign the entire HCA to a single virtual machine, enabling InfiniBand connectivity without using SR-IOV. The main

[6]Mellanox OFED - http://www.mellanox.com/page/products_dyn?product_family=26

drawback of such approach is no support for running multiple InfiniBand guests concurrently on a single compute node. For above reasons, this paper focuses on SR-IOV based approach. More information on the direct PCIe passthrough performance compared to SR-IOV can be found from Lockwood's blog [9].

Both resources at NCI and at A*CRC were provisioned by using the OpenStack interface to setup both environments. A major part of the setup effort was for OpenStack to play nicely with the InfiniBand interfaces. To make this possible, three additional modules were installed: A custom virtual interface module adds support for SR-IOV virtual function networking in the nova-compute component. An embedded switch module implements linking virtual functions to guests and enforces network access restrictions. A custom DHCP server adds InfiniBand support. On top of this, a few OpenStack out-of-tree patches were necessary in order to force the use of a single partition key, as required by the InfiniBand range extenders. After installing the additional modules and patches, compute nodes are configured to directly connect the HCA to the upstream network, bypassing the layer two agent traditionally present on OpenStack compute nodes — this functionality is now provided by the embedded switch.

In addition for Haswell servers, CPU passthrough was enforced instead of OpenStack defaulting to the Nehalem CPU architecture. This resulted in a 3-fold speedup in Linpack performance due to the AVX, AVX2 and FMA feature flags present in Haswell over Nehalem.

In addition to the two OpenStack provisioned setups, we utilised an existing non-OpenStack virtual cluster at A*CRC that was already setup using virt-manager and hosts a BeeGFS parallel filesystem [5]. This was used to test the parallel filesystem's performance using both native and virtual metadata & storage target backends on Haswell hardware and was also used for MPI micro-benchmarks.

Each cluster was interconnected to a Mellanox SX-6036 36-port switch and all servers utilised KVM/QEMU as the virtual machine monitor.

2.2. Benchmarking and VM configuration

This subsection details each benchmarking application used in this exercise we used together with the VM configuration for each one where appropriate. To present the possible worst-case scenarios, the highest recorded benchmarks out of several runs on native BM hardware were used whilst the lowest for VMs were recorded. The exceptions are the MPI and storage benchmarks, where we reported the average values.

2.2.1. iperf TCP performance

iperf was used to test the TCP bandwidth available between a pair of nodes. The test was multi-threaded, utilising all cores available on each server (24 on Haswell and 16 on SandyBridge) to saturate the available bandwidth. For the virtual test, each node hosted a single VM with all available cores allocated. The aggregate bandwidth achieved was recorded at the end.

2.2.2. InfiniBand write performance and latency

The *ib_write_bw* and *ib_write_lat*, part of the OFED perftools package, were used to test the RDMA bandwidth performance and latency of the InfiniBand interconnect in both native and virtual instances. The server and VM setup was the same as that for the iperf test. For the bandwidth and latency tests, 64k and 2-byte messages transfers were used respectively.

2.2.3. Linpack

Used to rank supercomputer installations in the Top500, the Linpack benchmark is used to ascertain the performance of a typical HPC application involving computation and communication by solving a dense linear system of equations [4]. Two Linpack benchmark types were used:

- A local Linpack application, namely the Intel Optimized SMP Linpack binary, was used to test on-node performance by solving a problem size with leading dimension of 60k elements. The virtual test used one VM with all cores allocated.
- An MPI-distributed Linpack, namely HPL, was tested on a pair of BM and VM servers. In this case, one MPI process per node was used with multi-threading enabled to utilise all the available cores available within each node. Hence for VM testing, one VM communicated with the other on distinct nodes, giving an idea of the communication performance through the IB interconnect in a virtual setting. A problem size with leading dimension of 120k elements was used together with a blocking size of 168 elements.

2.2.4. MPI Ping-Pong, Alltoall and Barrier microbenchmarks

Using the Intel MPI Benchmarks v4.0.0 package[7], we looked at the performance of message-exchange for a range of message sizes using the Ping-Pong test and the performance of MPI collective operations that involve the synchronisation of many MPI processes on Haswell hardware.

For this benchmark, three non-OpenStack servers were used where each VM was allocated one CPU socket of 12 cores and 64GB memory. The bare-metal test utilised the other, unallocated CPU socket and 64GB of memory available. Tab. 2 summarises the allocated resources.

Table 2. Summary of server specifications of the non-OpenStack A*CRC machines allocated to a VM or BM instance

VM Manager	virt-manager
CPU	Intel E5-2680v3 12-core Haswell (HW) Arch.
Memory	64GB 2133MHz DDR4
Storage	3x 512GB Micron M600 SSD 1x 1TB 10k RPM WD HDD
Network	Mellanox Connect-X3 FDR
Operating System	CentOS 6.5
# of servers	3: Arranged as 3 VM or BM instances, each with the resources stated above

The Ping-Pong test used one MPI process on two distinct nodes whilst all available cores were used for the MPI collectives with 36 MPI processes. The average latency or time to completion was recorded.

[7]Intel MPI Benchmarks - https://software.intel.com/en-us/articles/intel-mpi-benchmarks

2.2.5. Filesystem benchmarking with fs_mark

We used the fs_mark v3.3 benchmark utility[8] for this test to measure the rate of file creation on a given filesystem. This was executed via the Phoronix Test Suite[9] framework and the average result is specified in terms of number of files created per second. We looked at the performance of the BeeGFS parallel filesystem using the three non-OpenStack Haswell nodes each with 3 Micron SSDs. Each SSD is a storage target formatted with the XFS filesystem and a 16GB ext4 partition on one of the SSDs was used as a metadata target. The filesystem client was a spare bare-metal server that executed fs_mark on the filesystem mountpoint. Four benchmark tests were conducted:

- Creating 1000 1MB files using 16 threads with and without the use of sync/fsync.
- Creating 4000 1MB files spread across 30 subdirectories using 16 threads with and without the use of sync/fsync.

For conducting the native BM test, the SSDs were mounted outside the VM and the BeeGFS meta and storage software daemons were running natively whilst the virtual test involved attaching the block devices to the VMs and the BeeGFS daemons running inside the VM. In both cases, raw disk data mode was used i.e. the XFS/ext4 filesystems were readable when mounted outside the VM.

The stripe setting was set to one storage target with a chunksize of 512k bytes. Hence the individual 1M files are assigned to one SSD in a round robin fashion. With three metadata targets, the second test involving 30 subdirectories round-robins each subdirectory to a metadata target.

3. Results

Tab. 3 shows a summary of the benchmarks obtained on OpenStack-provisioned SandyBridge and Haswell hardware and also comparing both native and virtual environments to determine the total virtualisation overheads on both architectures. The filesystem benchmarks are recorded in tab. 4.

Table 3. Summary table of NCI-A*CRC InfiniCloud OpenStack performance benchmarks

Benchmark (units)	SB, native / virtual	HW, native / virtual
iperf (Gbits/s)	43.20 / 39.48	47.08 / 43.18
ib_write_bw (MB/s)	6003.99 / 5901.84	6075.36 / 5963.20
ib_write_lat (μs)	0.94 / 1.43	0.88 / 1.30
Local Linpack (Gflops)	279.15 / 268.41	779.45 / 654.39
MPI Linpack (Gflops)	506.02 / 476.11	1332.41 / 1329.18

3.1. Latency, bandwidth and linpack results on OpenStack InfiniCloud

Fig. 1 illustrates the IB RDMA write latency test and the measured virtualisation overhead for writing a 2-byte chunk of data. The overhead is slightly less on Haswell but both are relatively

[8]The fs_mark benchmark - http://sourceforge.net/projects/fsmark
[9]Phoronix Test Suite - http://www.phoronix-test-suite.com

Table 4. Summary table of parallel filesystem performance on file creation, comparing native and virtual storage backends

FS-Mark Benchmark	Native (files/s)	Virtual (files/s)
1k files	1992	981
1k files, no sync	4335	4168
4k files in 30 subdirs	2039	1172
4k files in 30 subdirs, no sync	3450	3074

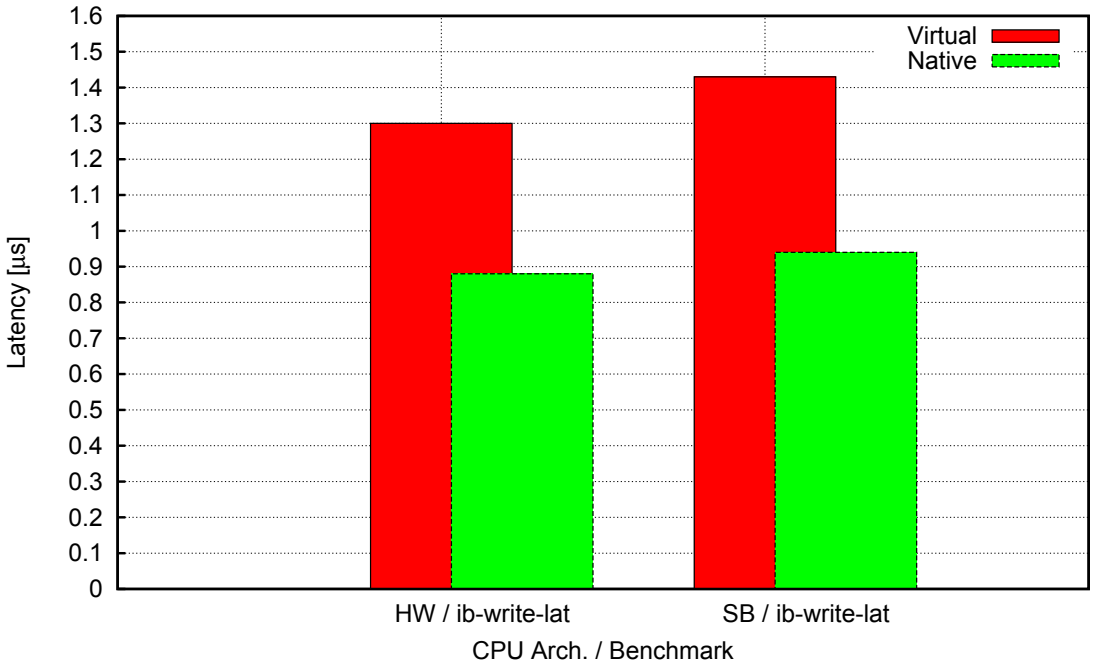

Figure 1. Bargraph showing the IB write latency measurements between native and virtual instances on both Intel architectures for 2-byte messages

significant when comparing native and virtual environments. It has been known from previous works that for small data sizes, the VM latency lags behind native latency [8, 9]. A possible reason is the way small messages are packaged in virtual functions.

When the IB RDMA write bandwidth is tested and shown in fig. 2, we see little overhead as we move to larger message sizes. Haswell is slightly ahead, although the noise encountered whilst executing the benchmark runs means any combination of VM/BM and CPU architecture can win. Since this benchmark is RDMA and should not involve much of the CPU, this shows the performance of the Mellanox interconnect and showing near identical performance between native and virtual instances.

For the TCP iperf test, the overheads for both architectures is greater than that for RDMA due to more involvement of the CPU in processing TCP packets and this illustrates the CPU virtualisation overhead as a result. We see that Haswell pulls ahead due to more processing power over SandyBridge.

For the local Linpack results in fig. 3, the Haswell virtual result shows around 84% performance relative to the native result. We believe that a large fraction of the overhead is due to the CPU virtualisation and this particular benchmark run was taxing the CPU cores. The CPU

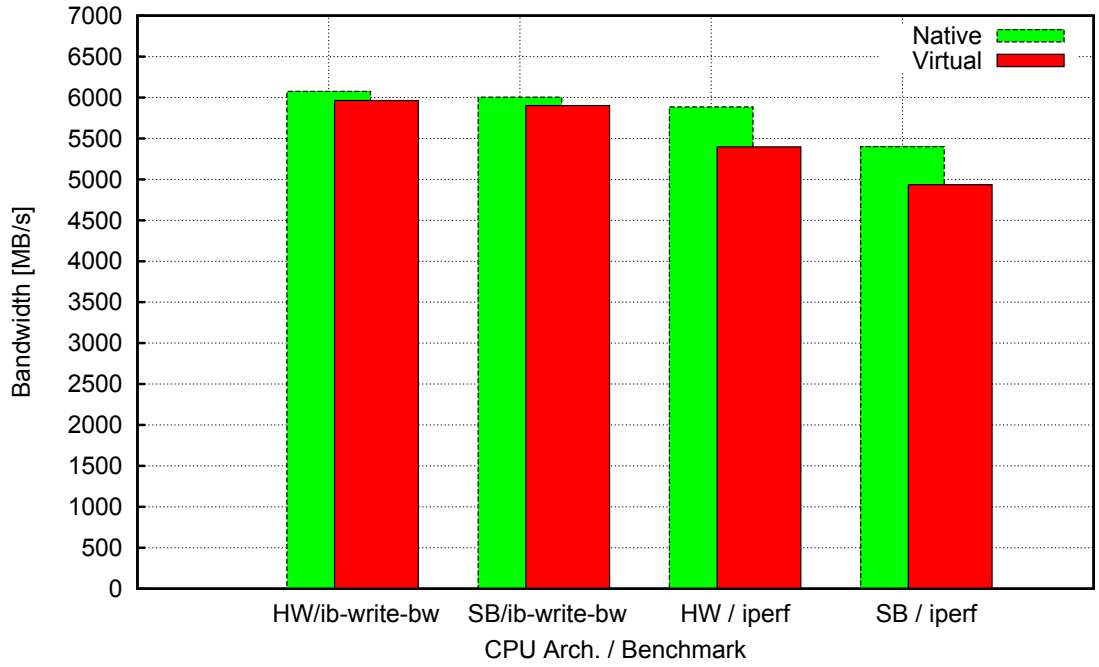

Figure 2. Bargraph showing the IB and TCP bandwidth measurements obtained using the ib_write_bw and iperf micro-benchmark programs respectively on both architectures

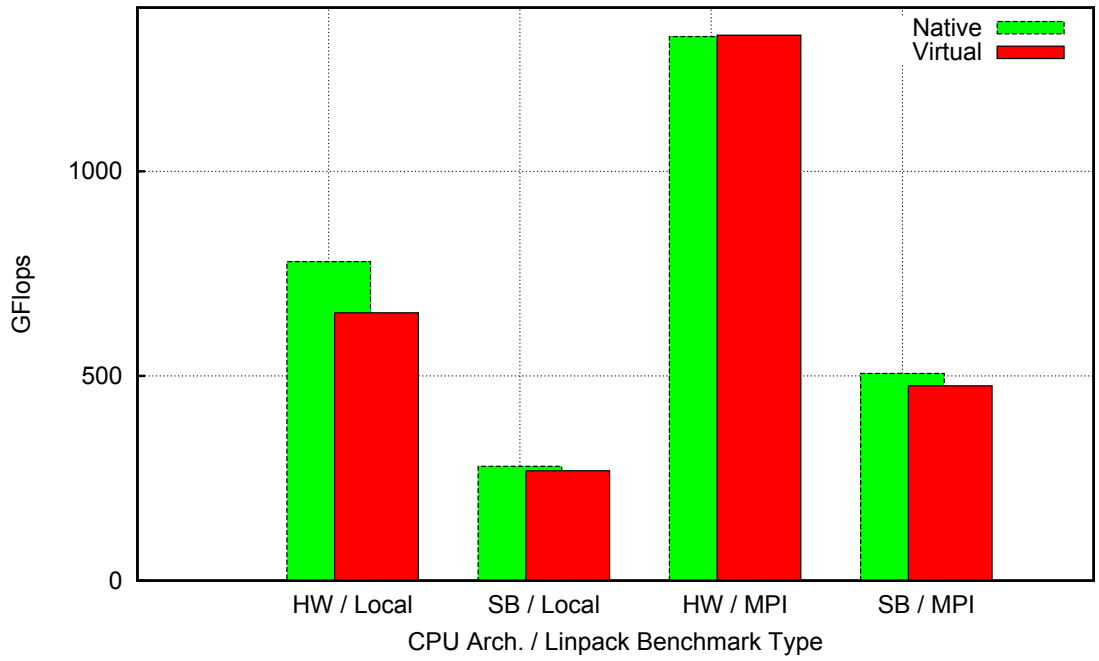

Figure 3. Bargraph showing the performance of both local and MPI Linpack on both CPU architectures

virtualisation overhead was found to be about 10% relative to native performance if no communication is involved [3]. For the other three results, little overhead is shown between virtual and native performance, showing near-identical performance for the communication part and it is likely all three cases were communication-bound. In the example of Haswell with MPI Linpack, more Haswell cores could complete the computational part more quickly hence more time

spent communicating. Comparing to the SandyBridge cases, users can expect more than double the performance improvement due to the superior Haswell architecture and more cores available. This should translate into comparable performance for real-world applications on Cloud computing platforms, as long as it is not heavy in collective communications as we illustrate next.

3.2. MPI microbenchmarks

When benchmarking the time taken to send a message back and forth between two processes, there is a difference for small messages until we reach 1M sized messages, shown in fig. 4. This is expected due to the lack of optimisation in the virtual functions packaging small messages, confirmed in previous studies [3], although we do not see any effect of inlining small messages in the native case. But the performance in virtual environments is far better than what is achieved using TCP on InfiniBand through IPoIB mode.

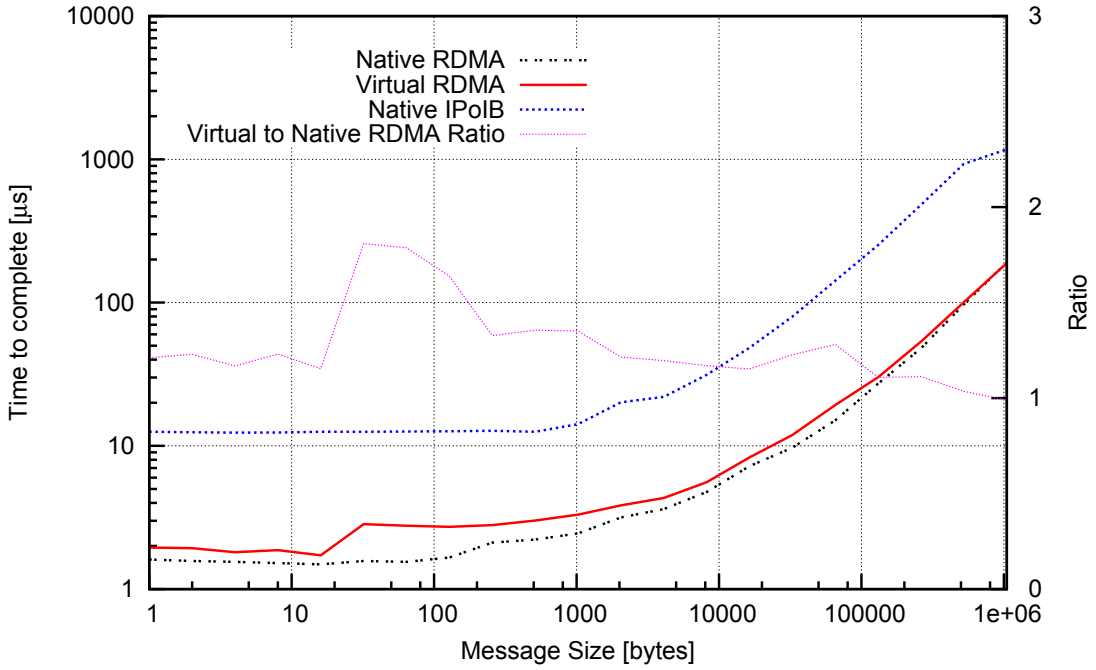

Figure 4. Graph showing a logarithmic plot of the time taken to complete a message pingpong between two MPI processes on distinct nodes against message sizes. The performance ratio between native and virtual RDMA is also shown

When testing MPI collective operations in fig. 5 and 6, we still see inferior performance compared to native mode, even on Haswell hardware. The time for all 36 MPI processes to sync to a barrier is 5.73μs compared to 20.92μs in VMs. For MPI Alltoall in fig. 6, the overheads increase the time by around 2.5 to 3 times, before settling around 1.2 times the native result for larger messages. The sharp jump in the ratio highlights the occurrence of the virtual result jumping to a higher completion time between message sizes 512 and 1024 bytes before the same phenomenon occurring in the native case between 1024 to 2048 bytes. This work confirms similar results from a detailed study on SR-IOV InfiniBand where collective operations are not as optimised on the virtual interfaces [8].

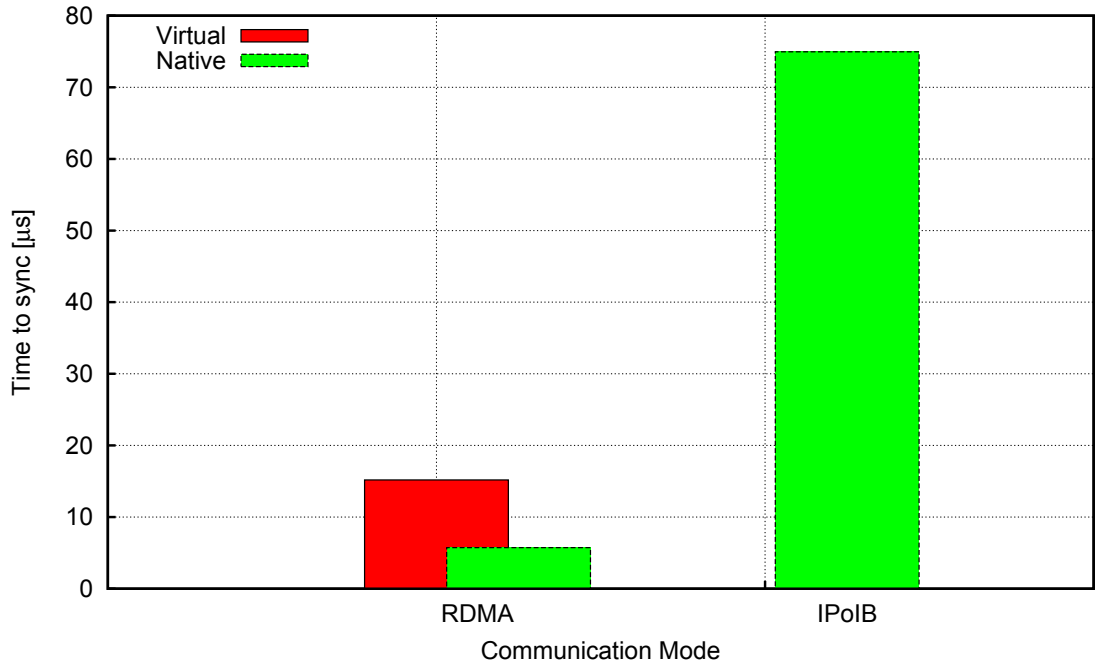

Figure 5. Graph showing the time to synchronise all 36 MPI processes to a collective MPI Barrier

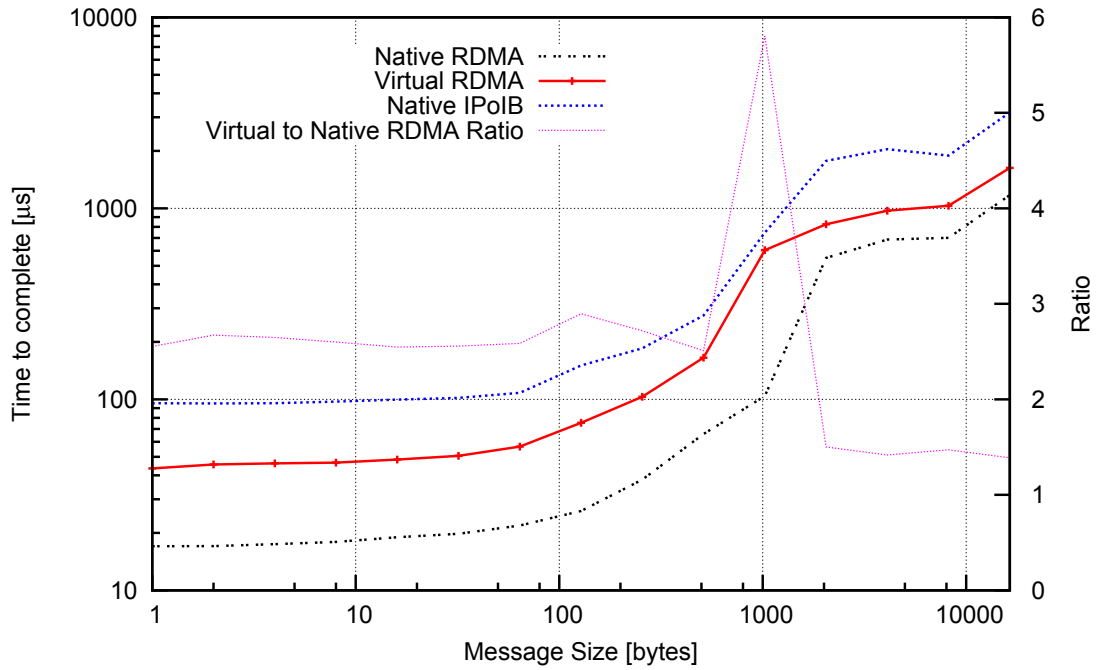

Figure 6. Graph showing a logarithmic plot of time taken to complete an MPI Alltoall collective amongst all 36 MPI processes against message sizes. The performance ratio between native and virtual RDMA is also shown

3.3. Parallel FS performance on native and virtual Haswell servers

Fig. 7 shows the BeeGFS parallel filesystem performance for native and virtual metadata and storage backends. When file syncing is enforced, the difference is about 50% whereas if no syncing is used, the filesystem performance is comparable between native and virtual. We

believe that there is further scope for filesystem tuning to improve performance as well as future improvements in the Mellanox virtual functions.

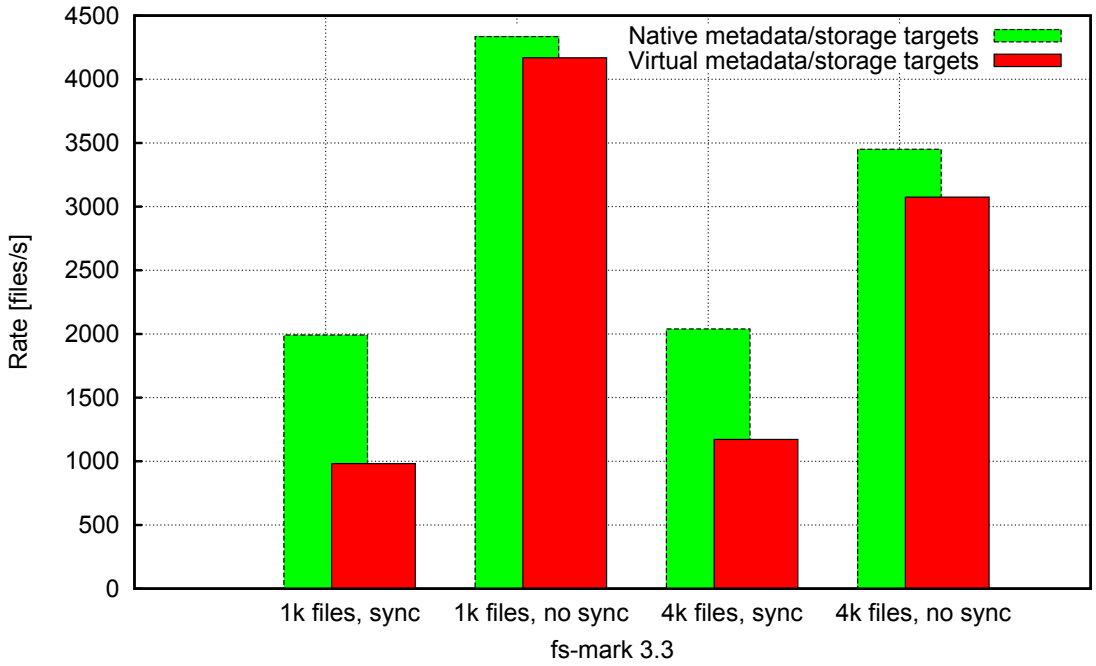

Figure 7. Graph showing the file creation rate of 1MB files, comparing native and virtual storage backends. The parallel filesystem is BeeGFS [5]

Conclusions

We have setup a pair of native and virtual nodes interconnected using SRIOV-enabled FDR InfiniBand and utilising SandyBridge hardware at NCI and Haswell over at A*CRC. These were provisioned using OpenStack with customised patches for InfiniBand and the suite of benchmark tests were conducted to test the network bandwidth, latency and application performance on native and virtual hosts. Later, a three node cluster was utilised to further test the native and virtual performance using MPI microbenchmarks and filesystem benchmarks. In summary, we found that:

- In terms of IB write bandwidth throughput, the difference is negligible for sufficiently large messages on both CPU architectures. For TCP bandwidth, there is an increased CPU virtualisation overheads on both architectures at around 9% with Haswell slightly increasing the throughput due to improved processing power.
- For IB write latency, we see an overhead of around 50 - 60% for 2-byte messages. This confirms previous work that virtual function interfaces are less-optimized for small messages. Haswell does seem to reduce the latency but the effect is minimal.
- For the local and MPI Linpack results, in three cases we see near-native performance. We believe this is due to the particular run not being CPU-bound and that the 168-element block setting ensured sufficiently large messages were exchanged to minimize the virtual overheads. The fourth case may indicate the run was CPU-bound and previous studies confirm a CPU virtualisation overhead of around 10% with no network message exchanges

involved. Clearly, A*CRC's Haswell nodes can offer InfiniCloud end-users a typical speedup of around 2.5 times over NCI's SandyBridge nodes.

- When using MPI collective benchmarks, a significant overhead exists when synchronizing MPI processes and this is still present on Haswell hardware, hence a network hardware limitation and already explored in previous studies.
- When examining parallel filesystem performance using native and virtual backends on Haswell servers and SSD storage, the difference is reduced when not enforcing sync or fsync, otherwise the performance difference is 2 times over for file creation rates. This is a quick look at performance and we believe there is room for filesystem and network parameter tuning.

Despite the overheads are involved, we believe that virtual environments are suitable for typical compute and I/O-intensive workloads whilst providing the benefits of software and resource management that virtualisation can offer. One such example on NCI-A*CRC's InfiniCloud platform is a genetic biological workflow [1] that can immediately take advantage of increased performance now from Haswell servers and from new technology in the future with little or no required adaptation of the software. In the future we would like to see continued effort in overhead reduction, especially for intensive, collective-based communication patterns common in scientific applications using FFTW for example. We believe that technologies such as Docker[10] and Linux Containers are an interesting proposition. Finally, it would be interesting to see how the performance varies on a larger HPC system and not restricted to the small prototype used.

*This work was supported by the A*STAR Computational Resource Centre and National Computational Infrastructure through the use of their high performance computing facilities.*

References

1. Kenneth Ban, Tin Wee Tan, Jakub Chrzeszczyk, Andrew Howard, and Dongyang Li. InfiniCloud: Leveraging Global InfiniCortex Fabric and OpenStack Cloud for Borderless High Performance Computing of Genomic Data and Beyond. Submitted to Supercomputing Frontiers 2015 conference proceedings, Singapore.

2. Jakub Chrzeszczyk, Muhammad Atif, Joseph Antony, Dongyang Li, Matthew Sanderson, and Allan Williams. Perspectives on implementation of a high performance scientific cloud backed by a 56G high speed interconnect. HPC Advisory Council Event, Singapore, `http://www.hpcadvisorycouncil.com/events/2014/singapore-workshop/preso/12_ANU.pdf`, November 2014.

3. Tiago Pais Pitta de Lacerda Ruivo, Gerard Bernabeu Altayo, Gabriele Garzoglio, Steven Timm, Hyun Woo Kim, Seo-Young Noh, and Ioan Raicu. Exploring infiniband hardware virtualization in opennebula towards efficient high-performance computing. In *Cluster, Cloud and Grid Computing (CCGrid), 2014 14th IEEE/ACM International Symposium on*, pages 943–948. IEEE, 2014. DOI: 10.1109/ccgrid.2014.90.

[10]Docker - https://www.docker.com

4. J Dongarra. Luszczek and A. Petitet (2001):" The LINPACK Benchmark: Past, Present and Future", University of Tennessee. Technical report, mimeo.

5. Jan Heichler. An introduction to BeeGFS. `http://www.beegfs.com/docs/Introduction_to_BeeGFS_by_ThinkParQ.pdf`, November 2014.

6. Marius Hillenbrand, Viktor Mauch, Jan Stoess, Konrad Miller, and Frank Bellosa. Virtual InfiniBand clusters for HPC clouds. In *Proceedings of the 2nd International Workshop on Cloud Computing Platforms*, page 9. ACM, 2012. DOI: 10.1145/2168697.2168706.

7. Wei Huang, Jiuxing Liu, Bulent Abali, and Dhabaleswar K Panda. A case for high performance computing with virtual machines. In *Proceedings of the 20th annual international conference on Supercomputing*, pages 125–134. ACM, 2006. DOI: 10.1145/1183401.1183421.

8. Jithin Jose, Mingzhe Li, Xiaoyi Lu, Krishna Chaitanya Kandalla, Mark Daniel Arnold, and Dhabaleswar K Panda. SR-IOV support for virtualization on infiniband clusters: Early experience. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pages 385–392. IEEE, 2013. DOI: 10.1109/ccgrid.2013.76.

9. Glenn K. Lockwood. High-Performance Virtualization: SR-IOV and InfiniBand. `http://glennklockwood.blogspot.sg/2013/12/high-performance-virtualization-sr-iov_14.html`.

10. Richard Moore, Luca Clementi, Dmitry Mishin, Phil Papadopoulos, Mahidhar Tatineni, and Rick Wagner. Comet: Realizing High-Performance. Virtualized Clusters using SR-IOV Technology. HPC Advisory Council Event, China, `http://www.hpcadvisorycouncil.com/events/2014/china-workshop/preso/3_Moore_Comet.pdf`, November 2014.

11. Tin Wee Tan, Dominic S.H. Chien, Yuefan Deng, Seng Lim, Sing-Wu Liou, Jonathan Low, Marek Michalewicz, Gabriel Noaje, Yves Poppe, and Geok Lian Tan. InfiniCortex: A path to reach Exascale concurrent supercomputing across the globe utilising trans-continental InfiniBand and Galaxy of Supercomputers. Submitted to Supercomputing Frontiers 2015 conference proceedings, Singapore.

Received July 3, 2015.