HPC in an OpenStack Environment

Joseph Stanfield and Nishanth Dandapanthula, June 2014

Introduction

As the concept of cloud computing continues to expand its market reach, many companies have discovered the advantage of encapsulating configuration details into a virtual setting allowing users to build a customized environment and run it in the cloud as the need arises for computational resources. Cloud computing would also seem to go hand-in-hand with a production HPC environment, offering virtually unlimited storage with instantly available and scalable resources. OpenStack is the open source cloud computing platform used in this study.

The applications in the HPC domain have massive requirements in terms of CPU, memory, I/O and interconnect. Traditionally HPC applications have been run on physical clusters, but with the trend moving towards cloud computing and virtualization, we wanted to see how these applications fare in a virtualized environment. In theory, the ability to scale out available resources on per-user basis would boost productivity and lower the total cost of ownership of the cluster. But, how does the performance of virtual machines (VM) compare to bare metal servers (BM)?

In this blog, we've set out to compare the performance of a physical server with a bare metal installation and a virtual machine, using a single node in similar environment~~s~~, with identical resources. The bare metal machine is a physical server with just a minimal OS installed. VM refers to the virtual machine running on a hypervisor on this bare metal machine using all the cores and memory of the bare metal system, thus both having the same configuration.

Consider a scenario with multiple project development needs, where users require a range of custom platforms for their individual projects. There may be a need for a whole server or multiple servers for various reasons such as application development, code beta testing, sharing a stable and uniform platform among collaborators etc. An administrator would be able to easily deploy an environment tailored to each user without having to re-provision the entire server farm for each project. Once the user is done, the VM's data or the VM itself can be archived for future use.

We study the differences in performance and the overhead raising from the use of VMS when compared to BMs in an HPC space. We present analytical results and weigh the pros and cons of each approach.

This is the first in a series of blogs where we will evaluate virtual machines, Linux containers, and bare metal servers and their respective tuning options from the perspective of applications in HPC domain. In future posts, we will expand this study at scale by introducing the interconnect component.

The test bed has a head node and a compute node with bare metal installations of RedHat Enterprise Linux 6.5. We installed RDO OpenStack on the head node and used that to add the compute node to the resource pool. The VMs are deployed solely on the compute node. The details of the test bed and the

BIOS configuration used are shown in Table 1. The BIOS settings chosen for this study on the bare metal machine are typical HPC recommendations for optimal performance [3~~10~~].

Table 1 Test Bed Configuration

Head Node		
Server	Dell PowerEdge R720	
Memory	16 x 8GB @ 1866 MHz	
Processor	2 x Intel Xeon E5-2697 v2 @ 2.70GHz	
OS	RHEL 6.5	
OpenStack Deployment	Icehouse-3 RDO PackStack	
Bare metal Node / VM node		
Server	Dell PowerEdge C6220 II	
Processor	2 x Intel Xeon E5-2680 v2 @ 2.8GHz	
Memory	8 x 16GB @ 1866 MHz	
BIOS	Version	2.1.2
	Turbo	Enabled
	C States	Disabled
	Hyper Threading	Disabled
	Node Interleaving (NI)	Disabled / Enabled
	System Profile	Max Performance
OS	RHEL 6.5	
Interconnect	Gigabit Ethernet between the head node and the compute node	
Hypervisor	QEMU KVM	

Performance and analysis

We used a sample set of applications from the HPC domain, both proprietary and open source for this comparison. The details of the applications are mentioned in table 2.

Table 2 Applications

Applications	Application Characteristics	Benchmark dataset used	Metric
HPL 2.0	Compute intensive benchmark measuring the floating point rate of execution	N = 110000, NB = 168	GFlops
ANSYS Fluent V15	Proprietary computational fluid dynamics application	Truck_poly_14m	Rating (Jobs/Day)
Stream Triad	Measures the sustained memory bandwidth	N = 160000000	MB/s
NAS Parallel Benchmark suite 3.3.1 **(NPB)**	Kernels and benchmarks derived from computational Fluid dynamics (CFD) applications	Class D	Rating (Jobs/Day)
LS-DYNA 6.1.0	Proprietary, structural and fluid analysis simulation software used in	Top crunch 3 vehicle collision	Rating (Jobs/Day)

Field Code Changed

	manufacturing, crash testing, aerospace industry, automobile industry etc.		
WRF 3.3	Open source application which helps in atmospheric research and forecasting	Conus 12KM	Rating (Jobs/Day)
MILC 7.6.1	MIMD lattice computation is an open source quantum chromo dynamics code which performs large scale numerical simulations to study the strong interactions which occur in sub atomic physics	Input file from Intel Corp.	Rating (Jobs/Day)

We evaluate four configurations here using the bare metal machine and the virtual machine while toggling the NI option in the BIOS.

1. Bare metal node with NI disabled (BM-1)
2. VM running on BM-1 (VM-1)
3. Bare metal node with NI enabled (BM-2)
4. VM running on BM-2 (VM-2)

Stream measures the sustained memory bandwidth using benchmarks such as Copy, Sum, Scale and Triad. We use the Triad benchmark since it includes the FMA (Fused Multiply Add operations) and it builds on all the other three benchmarks. It is also the closest synthetic benchmark to have an impact on application performance since FMA ops are a very important part of all the basic computations such as matrix based computations, vector dot products, Fourier transforms, polynomial evaluations etc. which in turn are used in most HPC applications.

Figure 1 shows the results of the Stream Triad memory bandwidth test. There is a ~26% drop on the system level memory bandwidth for VM-1 when compared to BM-1. This can be attributed to the hypervisor not being NUMA aware in terms of memory. We pin the virtual cores on the guest OS to correspond to the physical cores on the BM. But the memory is allocated at the discretion of the hypervisor. On BM-1, there are two NUMA nodes with 10 cores and 64 GB of memory associated with each NUMA node. On BM-2, there is a single NUMA node with all the cores and memory on the physical machine allocated to it. But on VM-1 and VM-2, we see a single NUMA node with all the cores and memory associated with it irrespective of the NI setting on BM.

The performance drop from BM-2 to VM-2 is ~1.5%. This is because of the fact that the underlying memory is interleaved among the two sockets and the memory access is uniform. But the drop from BM-1 to VM-2 is ~19% which is better than the 26 % drop from BM-1 to VM-1. This indicates that NI enabled may be the best option for the Guest VM in terms of memory bandwidth. For the purpose of this study we used the default configuration for memory management on the hypervisors. In upcoming blogs we will dig deeper into optimization options such as controlling the NUMA affinity of the guest machine.

Figure 1 Stream Triad memory Bandwidth

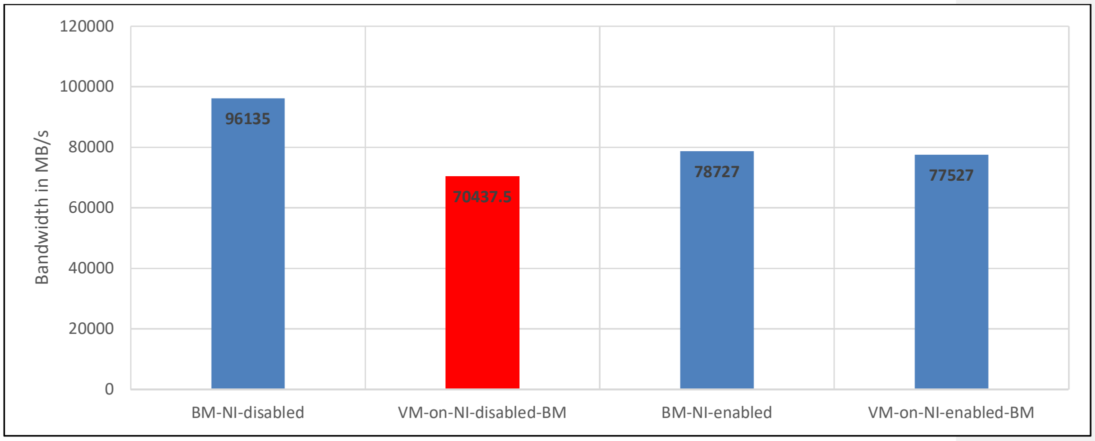


Figure 2 and Figure 3 show the performance of the applications listed in Table 1. The graphs show VM-1 and VM-2 performance relative to BM-1 performance. Most of the following applications are impacted by the CPU characteristics such as core speed, core count etc. and memory bandwidth requirements. Since the core speed and the core count are similar for all the configurations, the major deciding factor falls back to memory bandwidth. Consider all the applications shown in Figures 2 and 3 except for NPB's CG and IS and HPL. The performance on VM-2 is better than or equal to the performance of VM-1. This can be attributed to the higher memory bandwidth of VM-2 configuration compared to VM-1 as shown in Figure 1. All these applications perform 1% to 14 % lower on VM-2 when compared to BM-1.

NPB's IS is an integer sort benchmark and CG is a conjugate gradient benchmark. Both these benchmarks primarily perform a lot of random, irregular memory accesses and are highly dependent on memory bandwidth. They perform ~24% and 27% lower on VM-1 and VM-2 respectively than on the BM-1. NI enabled is adversely affecting these two benchmarks. HPL being a compute intensive workload should not be affected by memory bandwidth. But the studies shown here depict that VM-2 performs 11% lower than BM-1. The performance difference between VM-1 and BM-1 is ~5 %. This is not considerably high as a part of the lower performance can be attributed to the minor portion of the server's compute resources allocated to running the hypervisor on the BM. We are looking into the lower performance of VM-2. Please let us know regarding your suggestions if any in the comments section.

Figure *2* NPB performance relative to BM

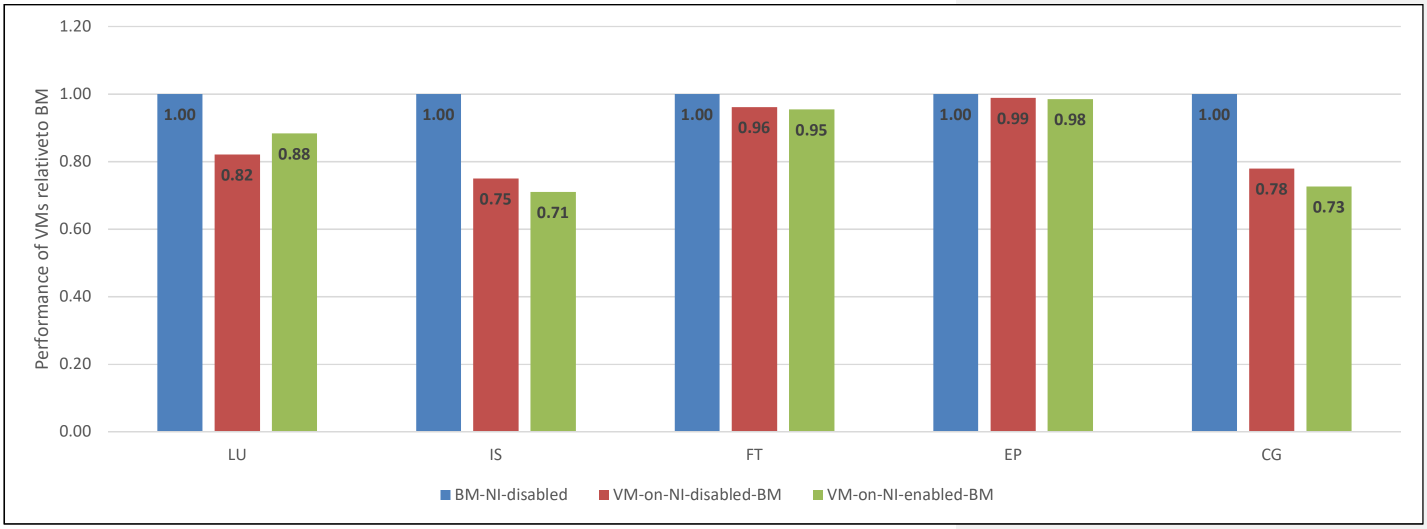
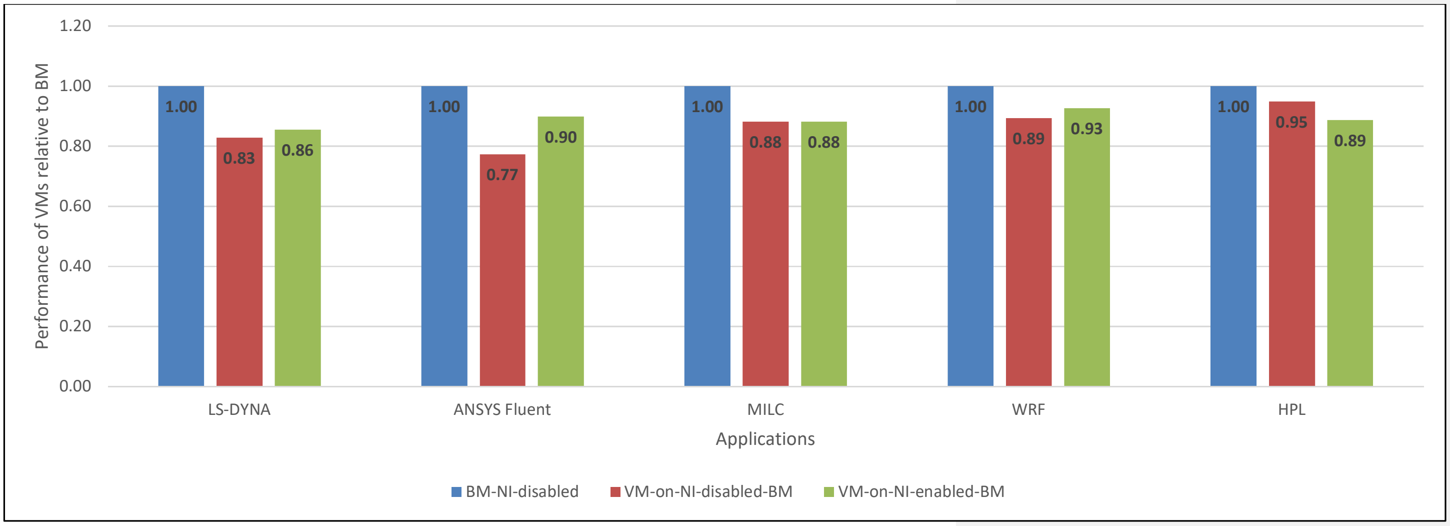


Figure 3 Application performance relative to BM



Conclusion

From the results above, certain HPC applications which are embarrassingly parallel and compute intensive may perform 1-2% lower on the VM relative to the BM, whereas applications which have very high memory bandwidth requirements may perform up to 25% lower on the VM relative to the BM. As mentioned previously this is all contingent on the application's requirements such as computation, communication (Interconnect), memory bandwidth etc. These applications are customized and tuned to reap maximum performance from the physical layer of the hardware. With cloud, performance may take a hit in terms of computational performance and I/O.

Alternatively, in some scenarios, a 25 % drop in performance may be acceptable considering the flexibility and convenience cloud computing provides. A cloud based environment would be a viable economic alternative to purchasing hardware. They provide instant availability, scalable resources, and software choices based on a user's needs. Thus a careful analysis of the application's requirements would need to be considered to determine the effectiveness of such an environment.

In conclusion, depending on the resource demand and application needs, a cloud based HPC environment could be beneficial or limited towards a project

Work in progress

From [3], we know that enabling logical processor / hyper threading in the BIOS impacts every application in its own way. In our upcoming blogs, we plan on studying the effects of enabling hyper threading on these applications on the VM. Another concept which is a contender to hypervisor virtualization is container based virtualization. We plan on studying how Linux Containers (LXC / Docker) perform when compared to VMs in an HPC environment.

The current study is based on a single node and works for some test cases. We want to take this study to a multi node level and introduce interconnect and scalability factors to study the dependency of applications on them.

References

1. http://www.ansys.com/staticassets/ANSYS/staticassets/resourcelibrary/presentation/ITSolutionsWebcast-ANSYS-IBM-April2012-REV6.pdf
2. http://i.dell.com/sites/content/business/solutions/whitepapers/en/Documents/11g-memory-selection-guidelines.pdf
3. ftp://ftp.dell.com/Manuals/all-products/esuprt_ser_stor_net/esuprt_poweredge/poweredge-1655mc_White%20Papers12_en-us.pdf