DOI 10.7603/s40601-014-0020-x

Virtual Machine Performance upon Intensive Computations

B.O. Yenké, A. A. Abba Ari, C. Dibamou Mbeuyo, and D. A. Voundi

Received 28 Aug 2015 Accepted 17 Sep 2015

Abstract— **The Virtual Machine (VM) technology has increased considerably within the last two decades and the performance of VMs seems closest to the host machine. This has motivated the use of VM in High Performance Computing (HPC). Running a scientific computation on VMs can severely impact the performance of applications. In addition, choosing the proper VM to run a specific application while minimizing the lost of performance is not an easy task. However, several virtualization solutions have been proposed. This paper presents the result of a comparison of the performance of the most commonly used VMs solutions: OpenVz, Linux-Vserver, LXC, XEN, KVM and VMware ESXi. The performance of these VMs are evaluated with the *NAS benchmark, Lmbench, IOzone* and *Intel ® MPI Benchmark* taking into account the consumption of resources such as CPU, memory, latency, disk and network communication. The result shows that some virtualization solutions present better performance in consumption of some of the previously mentioned resources than others. This work aims at helping a scientist in selecting the VM suitable to compute a specific task without a significant overhead on applications performance.**

Keywords— Virtualization; Virtual Machine; Performance Evaluation; HPC.

I. INTRODUCTION

The VM technology allows one to bypass the constraints on the compatibility of the machine and on those concerning the hardware resources, thus offering a higher degree of portability and flexibility. It ensures the security of the OS, flexibility and cross-platform compatibility. Designed to solve the problems of sharing major components of a computer system often for commercial purposes [1], [2], [3], the technology of VM is increasingly used for HPC because VMs offer progressive performance almost close to real machines [4].

There are several virtualization techniques. Full virtualization technique allows unmodified operating system to run as guests in a VM. Hardware assisted virtualization consist of addition of hardware instructions in order to assist virtualization software. In paravirtualization, it is necessary to have a modified operating system, aware that it runs in a virtualized environment. The guest directly operates on the hardware of host computer. In operating system level virtualization, guest VMs are processes currently running within a general-purpose operating system that has been modified to provide separate name spaces such that guests appear to be separate machines [5].

To simplify the exploitation of HPC environments and systems through virtualization technology, the virtualization platforms should provide excellent performance without considerable effort. Today, VMs are increasingly used to make high performance computing because they provide a consistent runtime environment [4], [6], [7], [8], [9]. Various VMs solutions whether proprietary or free are proposed. This, therefore, raises the question of choosing a VM solution instead of another according to the criteria of performance expected by HPC applications to be computed. For instance, among the various VMs proposed, which one gives a better performance in CPU consumption?

This paper sets out to present some elements allowing to choose a VM solution that matches with the expected performance of scientific computation. In so doing, we proceed to single host and cluster evaluation. For single host, we examine CPU with NAS Parallel Benchmarks, memory operations and latency with Lmbench suite and disk read/write with IOzone. Concerning cluster evaluation, we use several scenario of test by varying VM number in order to highlight network performance and scalability of each virtualization solution. To achieve that Intel ® MPI Benchmark were used.

The rest of this paper is organized as follows. Section II presents an overview of virtualization technologies and common problem of choosing a virtualization solution for HPC is highlighted in Section III. Section IV describes our experimental environment. The results obtained from the various experiments are discussed in Section V. In section VI, related works are presented. Section VII concludes the paper.

DOI: 10.5176/2251-3043_4.3.336

Published online: 13 December 2015

II. Virtualization Technologies

Virtualization's concept refers to techniques which allow running in the same physical machine, several operating system instances that share the same hardware. In such context, the installed systems are commonly called VMs. VM is running through a software layer called Hypervisor or Virtual Machine Monitor (VMM).

In general, virtualization solutions are implemented over one of the suitable four techniques: operating system level virtualization; paravirtualization; full virtualization. In this paper, six virtualization solutions are used. Three container-based virtualization solutions (LXC, OVZ and Linux VServer) and three paravirtualization and hardware-assisted virtualization solutions (KVM, XEN and VMware ESXi). A great number of virtualization solutions are proposed but the problem of choosing a solution according to the impact on applications performance have not yet been addressed, which is the goal of this work.

III. Problem of choosing a Virtualization solution for HPC

HPC developers are facing one problem which concerns the choice of the appropriate virtualization technology in line with their application, since each technology comes with its strength and weaknesses. In addition, applications are very different, for example some need a lot of CPU while others require memory. HPC applications do not all have the same resource needs. For these applications running in virtual or physical environments, the needs of resources differ from one application to another. For example, an application can have more CPU resource needs; another one may need an environment that has a good memory management. In the context of HPC applications running in virtual environments, the choice of virtualization solution to use is often a problem. These solutions do not always have the same capacity to manage CPU, memory, I/O, ... If an application designed to run in a virtual environment does a lot of I/O operations, then it would be better to use a virtualization solution adapted to this context. Therefore, raises the question of choosing a virtualization solution suitable to compute a specific task without a significant overhead on applications performance.

IV. Test environment and Benchmarks solutions

Performance is the key issue when virtualization is invoked. One of the goals of virtualization is to allow an application running in a VM to have the same performance as running on native machine. Multiple VMs running on the same physical machine must share resources properly and be well isolated from each other. This section presents the experimental environment used for tests.

The first series of tests consists of single host VM resources tests. Our measurements are conducted on an Intel ® Core™ i3-2100 processor, DDR3 4GB, HVM module and an SATA 7 200 tr/min disk. The host machine is running a minimal Debian Squeeze distribution with *Linux 2.6.35 amd64 kernel*, excepted the VMware ESXi solution with has its own operating system. Each guest OS runs with the following characteristics: 2923 Mhz frequency processor, 256 MB of *RAM*, a *LVM* partition with *ext4* filesystem except the VMware ESXi solution which uses a different file system manager. On each guest OS is installed the base packages: *gcc, g++, fortran77-compiler, NAS benchmark, Lmbench* and *IOzone benchmark.* For VMs cluster environment tests, the experiments were conducted in a cluster of 3.06 GHz Dual-Core processors including virtualization support, 2GB of DDR3 memory, 320GB disk and a network card Broadcom Netlink Gigabit Ethernet. Each machine has a Debian Wheezy i386 minimal installed. Machines are connected via a Gigabit Ethernet Switch. Among the machines, one acts as the server on which was installed the following applications: *MPICH2, Taktuk, VNC server*, and *Intel ® MPI Benchmark suite*. The selected solutions are first installed and the VM deployed and configured, before proceeding to performance tests. The performance analysis is performed by deploying on VMs, one program which allows to stress the component that performance is analyzed. On each VM, Benchmarks are executed on the guest OS and the resulting data are collected for the performance analysis. Experiments were conducted with the most widely used benchmarks in HPC environments: *NAS parallel benchmarks* [10] for CPU utilization; *IOzone* [11] for the local *ext4* filesystem; *Lmbench* [12] for memory operation and latency test and *Intel ® MPI Benchmark* [13] for network evaluation in VMs cluster environments.

In fact, network evaluation in VMs cluster environments consist of using multi-exchange and multi-Bcast microbenchmarks of *Intel ® MPI Benchmark* suite. Multi-Exchange Benchmark measures the bandwidth and communication time between p processes through the network ($p \geq 2$). The process involves in the test, form groups of periodic communication chains. Each chain has the same number of processes. Processes that can not belong to any group are squeezed out of the test. During the test, each process of a chain sends two messages, of a certain size to its neighbors and receives simultaneously two messages of the same size from those same neighbors.

The Multi-Bcast benchmark measures the communication time of a broadcast operation between p processes. The processes involved in the tests are organized into group of k processes ($2 \leq k \leq p$). For each data size, each process in a group in turn sends a message (with the same size) to all the other processes in the same group. Processes that can not belong to any group are squeezed out of the test. Regardless of the virtualization

solution, each VM has 256 MB of memory, 4GB of *LVM* disk partition, a *swap* of 1GB and *ext4* as file system. For experimentation needs, each VM has the following programs installed: *gcc, g++, MPICH2, Taktuk, expect, VNC server* and *Intel ® MPI Benchmark*.

For all tests, the experiments were repeated at least ten times and the results were almost the same.

V. Experimental Results and Analysis

In this section, the results and interpretations of performance analysis conducted are presented. For single host VM resources tests, the experiments were carried out on six of the most used virtualization solutions: OpenVz, Linux-Vserver, LXC, XEN, KVM and VMware ESXi. The performance evaluation is achieved by testing, for each virtualization solution, the CPU use, memory management, latency time, and Networks performance.

For VMs cluster tests, experiments consist of evaluation of network performances in case of simultaneous communication in a cluster. In fact, while scaling up the number of hosted VMs to 2, 4 or 5 physical node, the average bandwidth or average time of communication obtained during simultaneous VMs communication is measured with *Exchange* and *Bcast* microbenchmark of *Intel ® MPI Benchmark suite*. The experiments were carried out on XEN and KVM in both para and full virtualization and also on OpenVz and LXC.

A. Single host VM resources tests

1) CPU utilization

To test the performance of VMs related to CPU utilization, we used two micro-benchmarks of NAS Parallel Benchmarks suite: LU and BT. It is the Gauss-Seidel and the tri-diagonal algorithms for solving linear systems. Matrices of the two micro-benchmarks have sizes 64^3 and 102^3 and the number of iterations is 200.

It can be seen on Figure 1 that the OS virtualization solution *LXC* presents good performance when running HPC applications that use a lot of CPU resources in VM environment, followed respectively by *OpenVz* and *Linux-Vserver*. However, for special reasons, it may be necessary to deploy the VM in paravirtualization mode. In this case, according to the obtained results, the *Xen* solution is highly recommended.

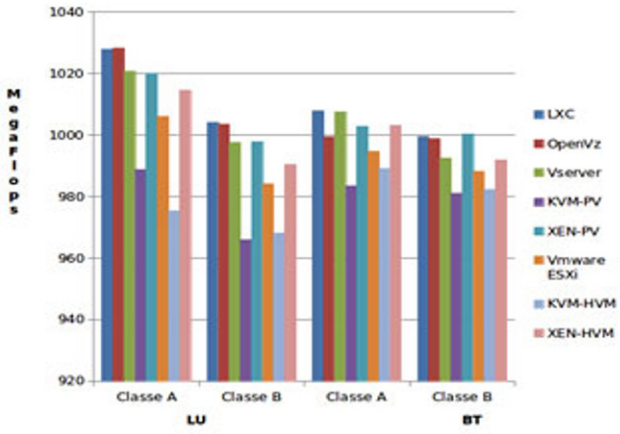

Figure1. Diagram of CPU utilization.

2) Memory operations

In this test, *lmbench* microbenchmarks suite are used for performance testing of VM regarding memory operations.

- **Processes creation.** The diagram in Figure 2 illustrates the differences between the measures obtained while taking the time of processes creation. It can be observed on Figure 2 that HPC applications making a lot of processes creation in virtual environments would benefit from the use of full virtualization solution VMware ESXi. It might be sometimes necessary to use OS virtualization solution. In this case, a good candidate OpenVZ followed by LXC.
- **Context switching.** For context switching it can be observed from Figure 3 that **Xen** and **KVM** are good solutions for both paravirtualization and full virtualization modes

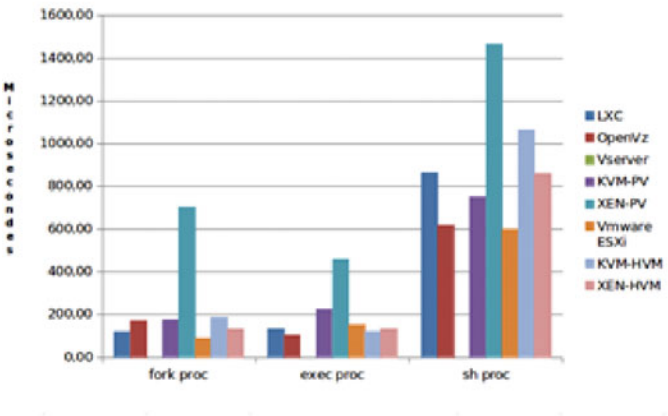

Figure 2. Processes creation.

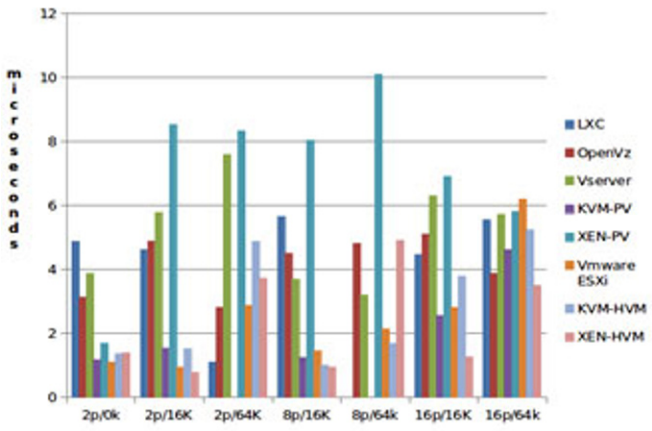

Figure 3. Context switching.

3) Memory latencies

The test scenario is the same as memory operations and benchmarks suite is also *lmbench*.

The diagram on Figure 4 presents the results of memory latencies.

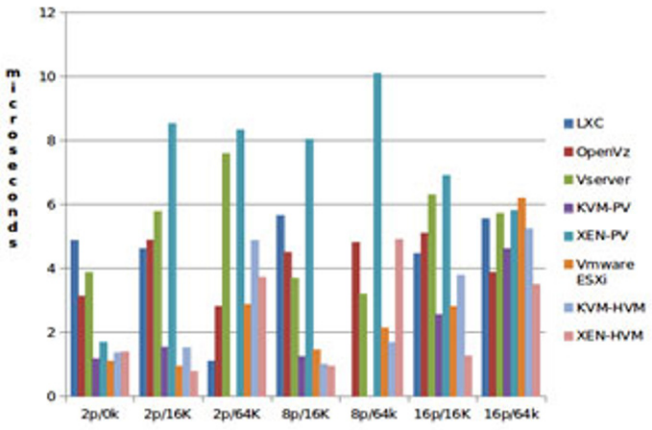

Figure 4. Memory latencies

From Figure 4, it can be observed that HPC applications running in clusters made up of VMs and requiring small latency time concerning memory and cache would benefit from the use of OS virtualization solutions: **Vserver, OpenVZ** and **LXC**. If for some reason the paravirtualization mode is indicated, a good virtualization solution is **Xen**.

4) Disk operations

Excepted VMware ESXi VMs (where the file system of the VM is actually a fixed-size file), the file systems of all our VMs are on *LVM* partitions and formatted in *ext4*. The performance analysis of our VM disk has been made with the tool *IOzone*. Transactions tested are: *read* and *write*. The file size tested in each case is 1GB, divided into several packages[1]; each packet is in turn divided into several sub-packets[2] and tested progressively. The results of this benchmark is normally presented in *3D* form according to file size, record size and read/write speed. For more readability reason, related cut in *2D* are given in Figures 5 and 6 for a better interpretation. The packet size is on the horizontal axis and the corresponding speed is on the vertical axis.

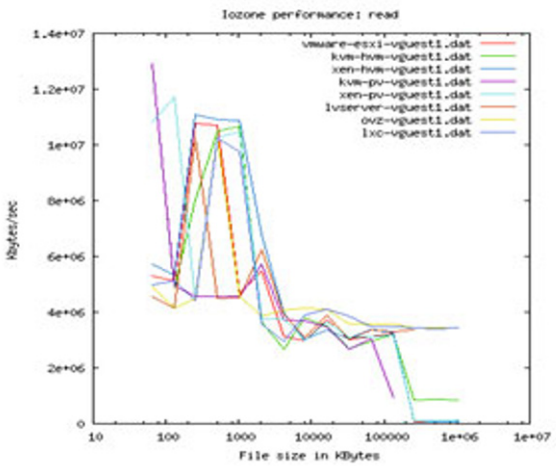

Figure 5. read

From Figure 5, it can be observed that HPC application making a lot of disk 5 operations would benefit from the use of: **KVM** and **Xen** in paravirtualization mode for files smaller than 100 KB; **Xen**, **VMware** and **KVM** in full virtualization mode for files in interval 100KB - 1MB; **Xen** in both two modes and **KVM** in only full virtualization mode; OS virtualization solution **OpenVz**, **LXC** and **Vserver** for files larger than 5 MB.

In order to compute HPC applications that do a lot of disk *write* operations in VM environments, according to curves in Figure 6, it would be very interesting to use: **KVM** in paravirtualization mode for files smaller than 100 KB; **Vserver** and **LXC** for file size in interval 100KB - 1MB and larger than 5MB; KVM in paravirtualization mode and **VMware ESXi** between 1MB and 5MB.

[1] The test is done gradually first 64KB, 128KB then, 256KB, ..., 1024KB.

[2] The sub-packets have size of 4KB, 8KB, 16KB, ..., 16384KB.

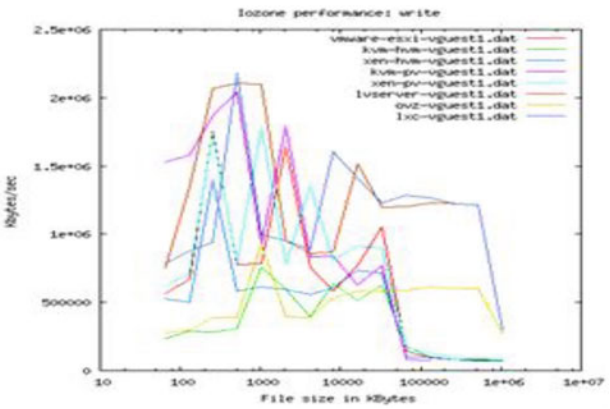

Figure 6. write

B. *VMs cluster environment tests*

1) *Network bandwidth*

Two tests scenarios were carried out with *Exchange benchmark* presented in section IV, in order to measure available bandwidth between $p \geq 2$ processes over the network of VM. In both two tests scenarios, experiments were conducted successively with network of 14 and 35 VMs. For the case of 14 VMs, 2 VMs per physical node were deployed and 5 VMs per physical machine for network of 35 VMs.

- **Scenario 1.** In the first scenario, the processes are organized in groups of two processes. In fact, the test is achieved by running the Exchange benchmark on 14 and 34 VMs respectively organized in 7 and 17 groups of 2 processes.

  Figures 7 and 8 present the obtained results in case of considered network. The results show that, the bandwidth achieved by all of the considered virtualization solutions is growing from *0* to *64KB*, before decreasing between *64KB* and *128KB* and approximately constant between *128KB* and *4MB*, with some exceptions in this interval. In all these cases, the bandwidth decreases with the growing of the number of VMs per physical machine. While with two VMs per physical machine, the bandwidth of some virtualization solutions is slightly above the threshold of *100KB/s* for certain sizes of data, with four VMs per physical machine is no more than *60KB/s* and with 5 VMs per physical machine we reached over *50KB/s*.

  OS-level virtualization **LXC** and **OVZ** realize the best performance with fairly similar performance. The performances of **KVM** in both paravirtualization and hardware assisted virtualization are also significant in case of two VMs per physical machines. When the number of VMs is greater than two, the performances of KVM become relatively bad compared to OS-level virtualization solutions. Xen records bad performance compared to the other virtualization solutions for every size of data and number of VMs per physical machines. Table 1 highlight the best solutions for this test scenario.

TABLE I. SCENARIO 1. EXCHANGE BENCHMARK.

	Number of VMs		
	14 *2VMs/node*		**34** *5VMs/node*
	Size of data		
	0o-16Ko	32Ko-4Mo	0o-4Mo
LXC	✓	✓	✓
OVZ	✓	✓	✓
KVMP		✓	
KVMH		✓	
XENP			
XENH			

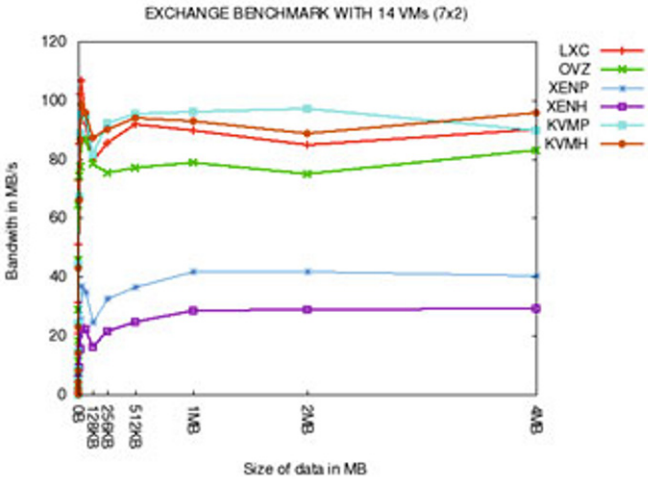

Figure 7. Scenario 1. Exchange benchmark with 14 VMs.

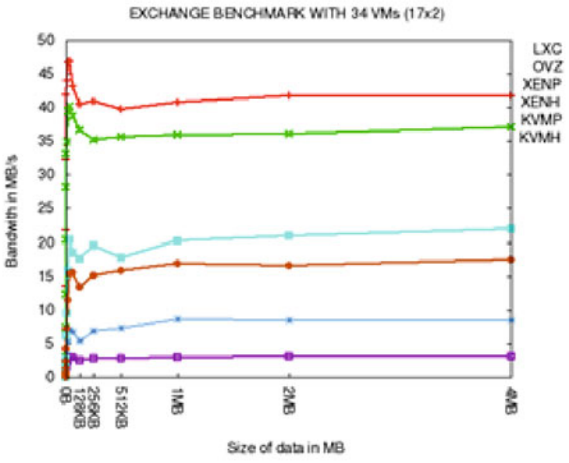

Figure 8. Scenario 1. Exchange benchmark with 34 VMs.

- **Scenario 2.** In the second scenario, the processes are grouped as single periodic chains. Figures 9 and 10 present the obtained results while running the

Exchange benchmark on 14 and 35 VMs organized into a single logical ring.

Despite some exceptions for certain sizes of data, the bandwidth carried by the OS-level virtualization solutions is growing from *0* to *64KB* and from *64KB* to *128KB*. Between *128KB* and *4MB*, the bandwidth is relatively stable. As in the previous scenario, the overall bandwidth decreases with the increasing of the number of VMs per physical machines. With 2 VMs per physical machine bandwidth of some virtualization solutions is slightly above the threshold of *100KB/s* for certain sizes of data and with 5 VMs per physical machines is not reached over *45KB/s*. However, it should be noted that the aggregated bandwidth of VM solutions in scenario 1 is greater than that of scenario 2.

According to presented results, **LXC** and **OVZ** realize the best performance with fairly similar performance for all sizes of data regardless of the number of VMs hosted per physical machines. Xen and KVM in both paravirtualization and hardware-assisted virtualization recorded bad performances with more than *40KB/s* data gaps for certain sizes, compared to OS-level virtualization solutions. Table II present a summary of the solutions that can bee recommended after this test scenario.

TABLE II. SCENARIO 2. EXCHANGE BENCHMARK.

	Number of VMs		
	14 *2VMs/node*		**35** *5VMs/node*
	Size of data		
	0o-16Ko	32Ko-4Mo	0o-4Mo
LXC	✓	✓	✓
OVZ	✓	✓	✓
KVMP			
KVMH			
XENP			
XENH			

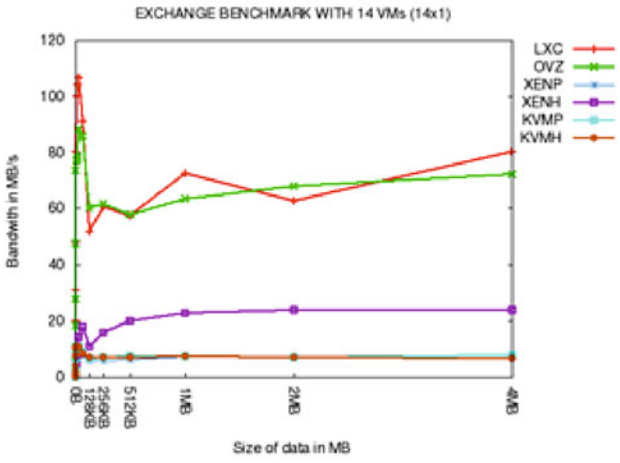

Figure 9. Scenario 2. Exchange benchmark with 14 VMs.

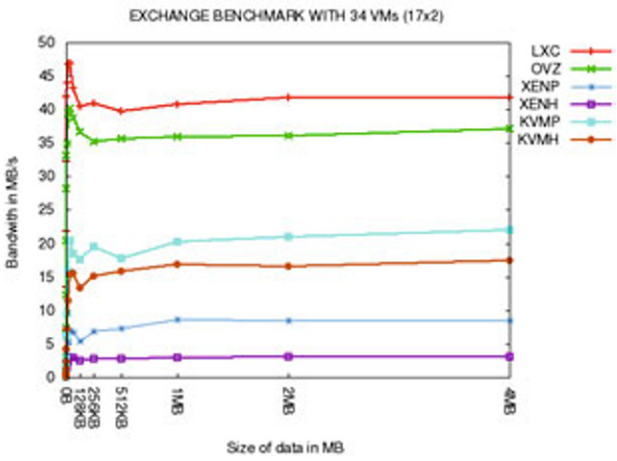

Figure 10. Scenario 2. Exchange benchmark with 34 VMs.

After these scenario of test, it's found that, bandwidth on a cluster depends on the virtualization solution, the number of VMs hosted per physical machines, the communication scheme between these VMs and the size of data exchanged in the network.

2) Communication time

As network bandwidth test, communication time is taken by performing two tests scenarios with Multi-Bcast Benchmark described in section IV. Also, for both two tests scenarios, experiments were achieved with 14 and 35 VMs with the same repartition of VMs per physical machines.

- **Scenario 1.** The first scenario is realized by organising the processes in groups of two. Bcast Benchmark is succesively deployed on 14 and 34 VMs respectively organized in 7 and 17 groups of 2 processes.

Table III present a summary of best solutions that can bee recommended after this test scenario. It appear that, OS-level virtualization solutions **LXC** and **OVZ** realize the best performance with fairly similar performance. **KVM**'s performance in both paravirtualization and hardware assisted virtualization is also appreciable. Xen, independently of considered kind of virtualization technique, achieve bad performance compared to the other virtualization solutions. Furthermore it should be noted that, in case of using 35 VMs, Xen in paravirtualization mode do not produce results.

TABLE III. SCENARIO 1. BCAST BENCHMARK.

	Number of VMs	
	14 *2VMs/node*	**35** *5VMs/node*
	Size of data	
	0o-4Mo	0o-4Mo
LXC	✓	✓
OVZ	✓	✓

KVMP	✓	✓
KVMH	✓	✓
XENP		*n/a*
XENH		

From Figures 11 and 12, it can been observed that, the communication time increases with the size of the data regardless the number of VMs involved. However, it should be noted that this time is more important when the number of VMs increase. Indeed, for the same size of data and any considered virtualization solutions, communication time increases with the number of VMs.

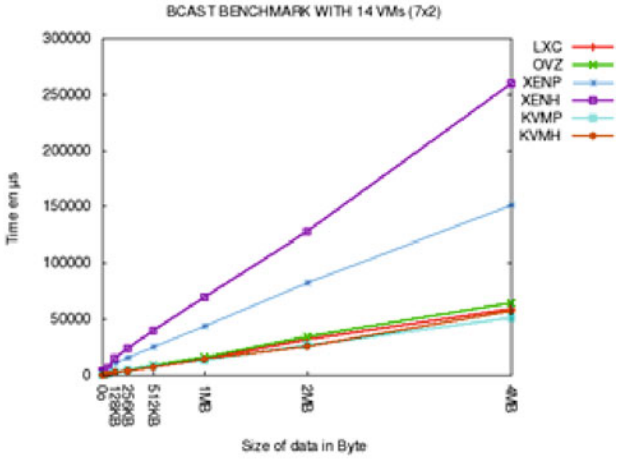

Figure 11. Scenario 1. Bcast Benchmark with 14 VMs.

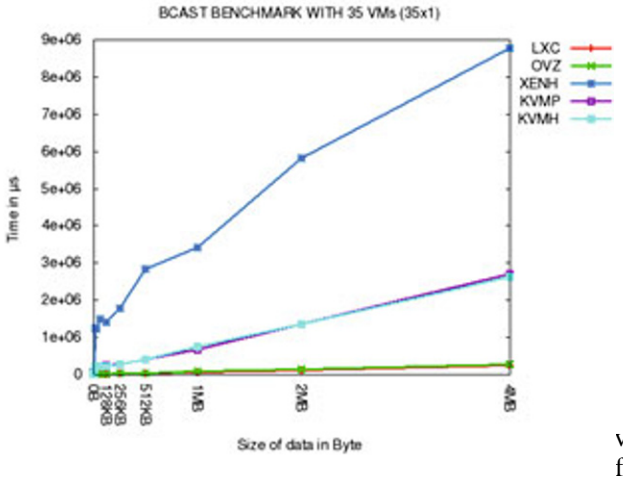

Figure 12. Scenario 1.Bcast Benchmark 35 VMs.

- **Scenario 2.** In this scenario, processes are grouped in single periodic chains and Bcast Benchmark is succesively launched with 14 and 35 VMs organized into a single logical ring.

As in scenario 1, the communication time increase with the size of data regardless of the number of virtual machines involved. However, it should be noted that this time increase with the number of VMs. Indeed for the same length of data and any considered virtualization solutions, communication time increases when the number of VMs per physical machine increases.

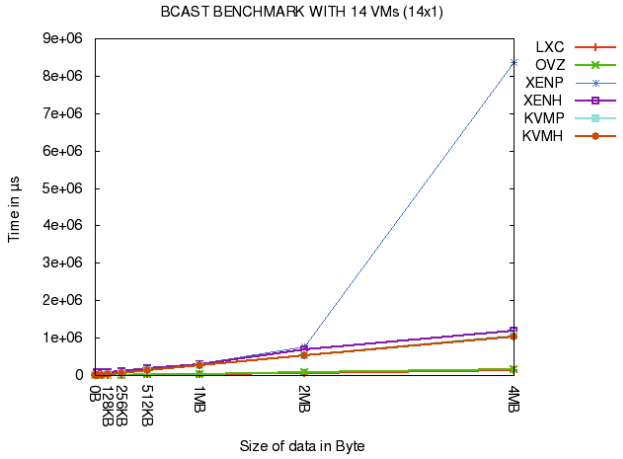

Figure 13. Scenario 2. Bcast Benchmark with 14 VMs.

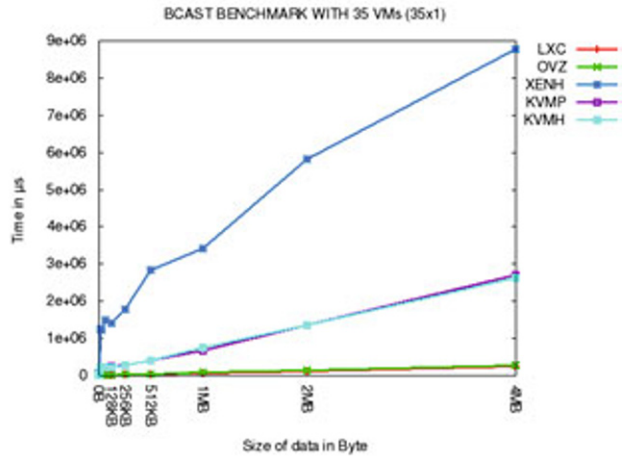

Figure 14. Scenario 2. Bcast Benchmark with 35 VMs.

From Figures 13 and 14 show that OS-level virtualization **LXC** and **OVZ** realize the best score with fairly similar performance. The performances of KVM both in paravirtualization and hardware assisted virtualization are also significant in case of two VMs per physical machines. When number of deployed VMs is greater than two per physical machines, KVM record bad performances. As in the first scenario, Xen independently of kind of virtualization achieve bad performance compared to the other virtualization solutions. It should also be noted that with 35 VMs per physical machine, not

available results have been found for Xen in paravirtualization mode. Table IV highlight the best solutions for this test scenario.

TABLE IV. SCENARIO 2. BCAST BENCHMARK.

	Number of VMs			
	14 *2VMs/node*		**35** *5VMs/node*	
	Size of data			
	0o-1Mo	2Mo-4Mo	0o-1Mo	2Mo-4Mo
LXC	✓	✓	✓	✓
OVZ	✓	✓	✓	✓
KVMP	✓	✓	✓	✓
KVMH	✓	✓	✓	✓
XENP	✓		*n/a*	*n/a*
XENH	✓	✓		

Generally, it is found that the communication time in a cluster depends on the virtualization solution, the number of VMs per physical machines, the communication scheme between VMs and the size of data exchanged between these VMs in the network.

VI. RELATED WORK

Several researches have been conducted in the performance evaluations of HPC applications running on native machine compared to those running on VMs. Some evaluation shows that HPC applications can achieve almost the same performance as those running in a native, non-virtualized environment [4], [13], [14].

In [9], four different HPC Linux Operating System Comparison with Xen-based kernel are presented. The work in [9] states that, in general, the Xen paravirtualizing system poses no statistically significant overhead over other OS configurations. The relative performance of native Linux, XenoLinux, VMware workstation 3.2 and User-Mode Linux is presented in [15]. Performance isolation benchmark that quantifies the degree to which a virtualization system limits the impact of a misbehaving virtual machine on other well-behaving VMs running on the same physical machine is presented in [5]. The conducted experiments include CPU, memory, disk, network intensive test and are based on VMware Workstation, Xen, OpenVZ and Solaris Containers. The results highlight differences between major classes of virtualization systems. Authors in [16] evaluate the trade-off between performance and isolation of container-based virtualization solutions Linux VServer, OpenVZ and Linux Containers, and they compared them with Xen hypervisor. They found that all used OS-level virtualization solutions have a near-native performance of CPU, memory, disk and network.

The work in [9] performs various test with: *SPEC CPU* suite to measure the performance of a system processor, memory system and compiler quality; Open Source Database Benchmark suite to measure multi-user Information Retrieval and On-Line Transaction Processing workloads of *PostgreSQL 7.1.3* database; *Dbench* to examine the throughput experienced by a single client performing around 90,000 file system operations; *SPEC WEB99* for evaluating web servers and the systems that host them; *Lmbench* which is operating system benchmarks to measure processes times, context switching times file and VM system latencies; and *ttcp* to measure bandwidth of network traffic both in transmit and receive.

In a VMs cluster environment, application performance is closely linked to its network performance. As shown in [17], applications that perform much communication are more affected in terms of performances in presence of virtualization. This implies necessity of measurement of network performance impact in VMs cluster environment. Authors in [18] reported a comparison in terms of network performance between native OS (running in the host machine) and three virtualization solutions including: Xen as paravirtualization solution, VMWare ESX as hardware assisted virtualization solution and OpenVZ like container based virtualization. *Netperf* [19] benchmark was used and the results show that OpenVZ achieve better latency close to native performance while in term of throughput, the performance of Xen is better. In the same order of idea, authors in [16] reported a comparison of four virtualization solutions, Xen in paravirtualization, OpenVZ, Linux-Vserver and LXC as container based virtualization. The test performed with benchmark *Netpipe* [20] between two processes in the network shows that container-based virtualization achieves better performance close to native machine.

Besides recording the performance of two processes hosted by VM which communicate via network, another important test is the evaluation of network performance in presence of many VMs running in the same node. In [21], evaluation of virtualization performance with Xen and KVM both in paravirtualization and hardware assisted virtualization is achieved. Tests are realized with *iperf* [22] while scaling up to either 2, 4 or 8 VMs hosted in a single physical node. Obtained results show that Xen and KVM achieve closely performance in terms of receiving and sending throughput. This fact highlights the advantage of paravirtualization solutions.

To complete these work, our paper first presents an evaluation of each VM solution according to CPU, memory, latency, disk and network utilization. Then, an analysis of virtualization in a cluster environment by scaling the number of VMs hosted on a single physical node, in order to evaluate impact of virtualization in such situation. This leads to know how throughput behaves in this kind of situation.

VII. Conclusion

This paper presented some criteria to be taken into consideration for performance evaluation of VM solutions. VMs have interesting properties implemented differently from one virtualization solution to another. In order to highlight the variation of performance of VMs in HPC environments, we conducted some experiments focused on resource consumption of resources such as CPU, memory, latency, disk and network. The performance of the most commonly used VMs solutions - OpenVz, Linux-Vserver, LXC, XEN, KVM and VMware ESXi - were evaluated with the *NAS benchmark*, *Lmbench*, *IOzone* and *Intel ® MPI Benchmark*. We have compared the experimental results obtained from the performance test on the VMs solutions used in regard to resource consumption. The originality of this work is that, in each case of test, we have recommended the best solution that can be used to build an efficient computing environment from VMs. Finally, this work aims at helping scientists to wisely choose a VM solution that matches with the expected performance of a scientific computation on VM cluster environments.

References

[1] R. J. Creasy, "The origin of the vm/370 time-sharing system," IBM Journal of Research and Development, vol. 25, no. 5, pp. 483–490, 1981.

[2] M. Intel, "Benchmark," 2008.

[3] S. Thibault and T. Deegan, "Improving performance by embedding HPC applications in lightweight xen domains," in Proceedings of the 2nd workshop on System-level virtualization for high performance computing. ACM, 2008, pp. 9–15.

[4] W. Huang, J. Liu, B. Abali, and D. K. Panda, "A case for high performance computing with virtual machines," in Proceedings of the 20th annual international conference on Supercomputing. ACM, 2006, pp. 125–134.

[5] J. N. Matthews, W. Hu, M. Hapuarachchi, T. Deshane, D. Dimatos, G. Hamilton, M. McCabe, and J. Owens, "Quantifying the performance isolation properties of virtualization systems," in Proceedings of the 2007 workshop on Experimental computer science. ACM, 2007, p. 6.

[6] M. F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis, "Virtualization for high-performance computing," ACM SIGOPS Operating Systems Review, vol. 40, no. 2, pp. 8–11, 2006.

[7] C. Engelmann, S. L. Scott, H. Ong, G. Vall´ ee, and T. Naughton, "Configurable virtualized system environments for high performance computing," in 1st workshop on system-level virtualization for high performance computing, Lisbon, Portugal. Citeseer, 2007.

[8] G. Vallee, T. Naughton, and S. L. Scott, "System management soft- ware for virtual environments," in Proceedings of the 4th international conference on Computing frontiers. ACM, 2007, pp. 153–160.

[9] L. Youseff, R. Wolski, B. Gorda, and C. Krintz, "Paravirtualization for HPC systems," in Frontiers of High Performance Computing and Networking–ISPA 2006 Workshops. Springer, 2006, pp. 474–486

[10] R. VanderWijngaart and B. A. Biegel, "Nas parallel benchmarks. 2.4," 2002.

[11] D. Capps and W. Norcott, "Iozone filesystem benchmark," 2008.

[12] L. W. McVoy, C. Staelin et al., "lmbench: Portable tools for performance analysis." in USENIX annual technical conference. San Diego, CA, USA, 1996, pp. 279–294.

[13] S. Thibault and T. Deegan, "Improving performance by embedding HPC applications in lightweight xen domains," in Proceedings of the 2nd workshop on System-level virtualization for high performance computing. ACM, 2008, pp. 9–15.

[14] R. McDougall and J. Anderson, "Virtualization performance: perspectives and challenges ahead," ACM SIGOPS Operating Systems Review, vol. 44, no. 4, pp. 40–56, 2010.

[15] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neuge-bauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," ACM SIGOPS Operating Systems Review, vol. 37, no. 5, pp. 164–177, 2003.

[16] M. G. Xavier, M. V. Neves, F. D. Rossi, T. C. Ferreto, T. Lange, and C. A. De Rose, "Performance evaluation of container-based virtualization for high performance computing environments," in Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on. IEEE, 2013, pp. 233–240.

[17] C. Macdonell and P. Lu, "Pragmatics of virtual machines for high performance computing: A quantitative study of basic overheads," in Proc. of the High Perf. Computing & Simulation Conf, 2007.

[18] V. Chaudhary, M. Cha, J. Walters, S. Guercio, and S. Gallo, "A comparison of virtualization technologies for hpc," in Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on. IEEE, 2008, pp. 861–868.

[19] R. Jones et al., "Netperf: a network performance benchmark," Information Networks Division, Hewlett-Packard Company, 1996.

[20] Q. O. Snell, A. R. Mikler, and J. L. Gustafson, "Netpipe: A network protocol independent performance evaluator," in IASTED International Conference on Intelligent Information Management and Systems, vol. 6. Washington, DC, USA), 1996.

[21] L. Nussbaum, F. Anhalt, O. Mornard, and J.-P. Gelas, "Linux-based virtualization for hpc clusters," in Montreal Linux Symposium, 2009.

[22] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "iperf: Testing the limits of your network," 2003.

AUTHORS' PROFILE

Dr. Blaise Omer YENKÉ is a Senior Lecturer and researcher in computer engineering. He is the Head of Department of Computer Engineering at the University Institute of Technology, University of Ngaoundéré, Cameroon. He received the Ph.D. degree in 2010 from the University of Yaoundé 1 in Cameroon and the University of Grenoble in France, in an international joint supervision. His current research interests include High Performance Computing, Distributed Systems, Fault Tolerance, Sensor Networks Design and Sensor's Architecture.

Ado Adamou ABBA ARI is an Assistant Lecturer in computer engineering at the University of Maroua, Cameroon. He received the B.Sc. degree in mathematics and computer science in 2010 and the M.Sc. degree in computer engineering in 2012 from the Faculty of Science, University of Ngaoundéré, Cameroon. He is now enrolled in an international joint supervision Ph.D. thesis program among the Université Paris-Saclay, France and the University of Ngaoundéré, Cameroon. His Ph.D. thesis research is focused on swarm intelligence based clustering and routing in wireless sensor networks.

Cyrille DIBAMOU MBEUYO received the B.Sc. degree in architecture of systems and networks in 2009 and the M.Sc. degree in computer engineering in 2012 from the Faculty of Science, University of Ngaoundéré, Cameroon. He is now enrolled in a Ph.D. thesis program in computer science at the University of Ngaoundéré. His researches are based on scheduling the checkpoint/restart of malleable tasks in virtual environments for High Performance Computing.

Donald Armel VOUNDI is a Teacher of Computer Science in a secondary school in Cameroon. He received the First Grade Secondary School Teacher Diploma in 2010 from the Higher Teacher Training College of the University of Yaoundé, the B.Sc. degree in computer science in 2011 from the Faculty of Science, University of Yaoundé 1 and the M.Sc. degree in computer engineering in 2014 from the Faculty of Science, University of Ngaoundéré, Cameroon.