The NAS Benchmark Kernels for Single and Multi-Tenant Cloud Instances with LXC/KVM

Anderson M. Maliszewski
Laboratory of Advanced Research on Cloud Computing (LARCC),
Três de Maio Educational Society (SETREM)
Três de Maio – RS – Brasil
andersonmaliszewski@gmail.com

Dalvan Griebler
Laboratory of Advanced Research on Cloud Computing (LARCC),
Três de Maio Educational Society (SETREM),
Pontifical Catholic University of Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil
dalvan.griebler@acad.pucrs.br

Claudio Schepke
Federal University of Pampa (UNIPAMPA)
Laboratory of Advanced Studies (LEA)
Alegrete – RS – Brasil

Alexander Ditter
Chair for Computer Architecture,
Friedrich-Alexander University Erlangen-Nürnberg (FAU),
Erlangen, Germany

Dietmar Fey
Chair for Computer Architecture,
Friedrich-Alexander University Erlangen-Nürnberg (FAU),
Erlangen, Germany

Luiz Gustavo Fernandes
Pontifical Catholic University of Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil

***Abstract*—Private IaaS clouds are an attractive environment for scientific workloads and applications. It provides advantages such as almost instantaneous availability of high-performance computing in a single node as well as compute clusters, easy access for researchers, and users that do not have access to conventional supercomputers. Furthermore, a cloud infrastructure provides elasticity and scalability to ensure and manage any software dependency on the system with no third-party dependency for researchers. However, one of the biggest challenges is to avoid significant performance degradation when migrating these applications from physical nodes to a cloud environment. Also, we lack more research investigations for multi-tenant cloud instances. In this paper, our goal is to perform a comparative performance evaluation of scientific applications with single and multi-tenancy cloud instances using KVM and LXC virtualization technologies under private cloud conditions. All analyses and evaluations were carried out based on NAS Benchmark kernels to simulate different types of workloads. We applied statistic significance tests to highlight the differences. The results have shown that applications running on LXC-based cloud instances outperform KVM-based cloud instances in 93.75% of the experiments w.r.t single tenant. Regarding multi-tenant, LXC instances outperform KVM instances in 45% of the results, where the performance differences were not as significant as expected.**

Index Terms*—Cloud Computing; High Performance Computing; Multi-tenancy; Benchmark; Infrastructure as a Service; Virtualization.

I. Introduction

Cloud computing architectures can generally be represented as a layered stack with increasing levels of abstraction. The first layer, the closest to the "bare metal" hardware is the hypervisor. It enables the shared utilization of resources on the high-level layers, such as *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS), and *Software as a Service* (SaaS) [1]. Although cloud computing technologies have been improved in the last decade [2], there is still a performance overhead involved in virtualization. Especially in the field of scientific computing, applications often require *High-Performance Computing* (HPC) capabilities [3], usually with low latency and high throughput network interconnections, hardware specific intrinsic instructions or *Direct Memory Access* (DMA). To better identify potential, virtualization induced, and performance bottlenecks, a proper and thorough performance analysis is paramount important before migrating applications to a cloud environment.

Performance analysis for HPC applications in cloud computing environments has been a current research problem. Our previous works [4] as well as related works (Section II) have tackled the problem in different aspects, goals, environments, and conditions. We found that the literature still lacks for empirical studies that address the comparison of container-based and kernel-based cloud instances with LXC/KVM virtualization technologies over private cloud environment conditions. Many other works create an evaluation about hypervisors and virtualization technologies. Likewise, the evaluation of private cloud environment is mainly done using OpenStack (most popular Cloud Management Platform). Consequently, our goal in this work is to deploy a private cloud with CloudStack supporting *Linux Containers* (LXC) and *Kernel-based Virtual Machine* (KVM) virtualizations with single and multi-tenants cloud instances. We also focus on multithreading parallelism using the OpenMP benchmark kernels developed by the *NASA Advanced Supercomputing Division* (NAS), the *NAS Parallel*

978-1-5386-7879-4/18/$31.00 ©2018 IEEE
DOI 10.1109/HPCS.2018.00066

Benchmarks (NPB)[1], which represent a wide range of scientific application classes. Moreover, we use statistic analysis to compare and validate our experimental results. The main contributions of this paper are summarized such as follows:

- A performance evaluation, analysis, and comparison of NAS Benchmark Kernels (scientific workloads) running on different types of private cloud instances;
- An evaluation of the private cloud infrastructure performance with single and multi-tenants cloud instances.
- A statistical view and analysis of the experiments.

The remaining of the paper is divided in six sections. Section II presents the related work. In Section III we present the experimental setup and methodology, along with the software and cloud environments. Section IV presents the background regarding the virtualization platforms and benchmark suite used in our evaluation. In Section V we compare the performance of our setups and present the respective results for each environment. Next, in Section VI, we statistically analyze our results and, lastly, we conclude in Section VII.

II. Related Work

Cloud computing evaluation became a hot topic in research field, and many papers have made performance evaluations [5], [6], [3], [7]. We consider a related work those with virtualization or cloud computing evaluation on scientific workloads.

In our previous performance analysis under private cloud conditions, we tackled different aspects. The environments were also the Cloudstack manager with two different types of cloud instances (KVM and LXC). In Vogel, et al. [6], the work focused on the network evaluation and optimization. Our findings demonstrated that KVM-based cloud instances have small network performance degradation regarding throughput. The container-based instances had the best results and KVM instances presented the worst latency. Griebler, et al. [8] made a performance analysis with PARSEC benchmark using the three mainstream application domains (financial, data mining, and media processing). Results highlighted that these applications in the LXC instances tend to outperform KVM when there is a dedicated machine resources environment. However, when two instances are sharing the same machine resources, applications tend to achieve better performance in the KVM instances. In this paper now, we used NAS Parallel Benchmark to represent the scientific applications with OpenMP.

For instance, Roloff, et al. leads to a detailed comparison of HPC applications running on three cloud providers (Amazon EC2, Microsoft Azure and Rackspace). The characteristics analyzed such as deployment facilities, performance and cost efficiency were listed and compared with clustering machines. They also performed their experiments using OpenMP and MPI version of NPB for Xen and Hyper-V in a OpenStack cloud environment. Results showed that HPC can work efficiently in the cloud, but the authors emphasize large differences between cloud providers, suggesting that behavior and application types perform differently depending on the cloud scenario [9]. In contrast, our paper is focusing on private cloud managed by CloudStack and provides a performance analysis between KVM and LXC technologies.

To evaluate the communication performance of HPC applications, a study was developed by Okada, et al. using the NAS MPI parallel benchmarks. They compared the execution behavior in Google Compute Engine, OpenStack using KVM virtualization, and a NUMA multiprocessor system using LXC container. The authors concluded that HPC users should use the appropriate number of vCPUs on each *Virtual Machine* (VM), avoiding the overcommitment of resources because application performance may be affected by scheduler and hyper-threading issues [10]. Differently, our paper focus on multithreading parallelism by using NAS OpenMP parallel benchmarks to compare LXC and KVM virtualization over private cloud conditions managed by CloudStack.

Furthermore, another performance evaluation was done by Xavier, et al., where various container-based virtualization experiments were performed (LXC, OpenVZ and Linux-VServer) for HPC. LINPACK, STREAM, IOzone, NetPIPE, NAS OpenMP and MPI parallel benchmarks and the *Isolation Benchmark Suite* (IBS) were used to evaluate performance, memory, disk, network, overhead and isolation, comparing containers with Xen virtualizations. The study highlights that HPC can only take advantage of virtualization if the overhead is reduced. It also argues that containers have almost native hardware performance, with differences between them in the implementation of resource management. However, container-based systems are not yet mature because they do not have better isolation, meanwhile, if HPC does not require resource sharing, a container can be attractive because of the minimal overhead [11]. Our work differs from this study in the virtualization technologies used and in the private cloud conditions.

Kang, et al. conducted an in depth system performance comparison using Amazon EC2 and Openstack with focus on KVM and Xen hypervisors. They evaluated CPU, memory, storage and network resources. Moreover, they compared and analyse diverse performance aspects based on hypervisor, storage, and network configurations. The benchmarks used were PerfKitBenchmarker, CoreMark, HPCC, SPEC CPU™2006, NPB, Fio, Iperf and Netperf. Using similar VMs instance types results on similar system performance for CPU and memory-intensive benchmarks. For storage, several AWS volume types were compared showed the improvements available using SSD volumes. Related to network performance, AWS heavily depends on the instance types, instead, OpenStack instances can have very high performance with little virtualization overhead, but can also be poor when vCPUs are being used. Therefore, when hardware configurations are similar AWS and OpenStack have corresponding performance. However, OpenStack has the advantage that it can accommodate more hardware heterogeneity due to its openness [12]. In contrast, our paper used Cloudstack as cloud management, LXC based-clouds and also perform a statistical comparison.

Moreover, J. Zhang, et al. compared the performance of KVM, Docker, and native environment as well as compar-

[1]https://www.nas.nasa.gov/publications/npb.html

TABLE I
SERVICE OFFER OF CLOUDSTACK INSTANCES

Name	Processor	GHz	vCPUs	Mem (GB)	Network
Single Tenant HPC	Xeon X5560	2.80	8	21.60	1GbE
Multi-Tenants HPC	Xeon X5560	2.80	4	10.80	1GbE

ing *Single Root I/O Virtualization* (SR-IOV) and PCI passthrough with high performance interconnects (InfiniBand). In addition, this paper used representative HPC applications and benchmarks such as Graph500, NAS, LAMMPS and SPEC MPI 2007. As contribution this work shows that container-based solution can deliver better performance than hypervisor-based solution overall. Finally, this evaluation also indicate that passthrough performs better than SR-IOV in almost all micro-benchmarks and applications. However, it can not be ignored the fact that SR-IOV is able to support multiple VMs. The combination of container with PCI passtrought is able to deliver near-native performance for end applications [13]. This work was done without a cloud management platform. In contrast, our paper introduces the comparison of LXC and KVM in a private cloud.

III. EXPERIMENTAL SETUP

This research investigates application performance in a virtualized cloud environment, where the configuration of hardware and software is intended to resemble a production environment with varying parameters. These variations include (i) the type of workload, i.e., the different types of benchmarks and applications of the NPB, (ii) the type of resource allocation (exclusive vs shared) and (iii) the type of hardware virtualization using LXC and KVM.

All tests were performed in the *Laboratory of Advanced Research on Cloud Computing* (LARCC)[2] at SETREM. The underlying software stack is based on CloudStack 4.8 as the cloud middleware and three identical servers running Ubuntu 14.04, KVM 2.0.0 and LXC 1.0.8. The hardware of each server is composed of one Intel Xeon X5560 quad-core 2.80 GHz processor with Hyperthreading intentionally disabled, 24GB RAM (DDR3 1333MHz), SATA II based storage and a 1GbE network connection. Software and benchmarks are compiled using the GNU Compiler Collection based Fortran compiler in version 4.8.5 (Red Hat 4.8.5-11). The general cloud structure consists of one front-end node for administration and management along with two compute nodes to run the workloads and applications. Both, primary and secondary storage, locations are exported from the front-end node via NFS and are used for storing the VM images, templates and operating system images. The service offer used on the LXC and KVM-based cloud instances are shown in the Table I. As a good practice the utilization of memory allocated (RAM) in the instances are 90% of the full node capacity.

Our methodology is depicted in Figure 1. The NAS OpenMP Parallel Benchmark was used and compiled with

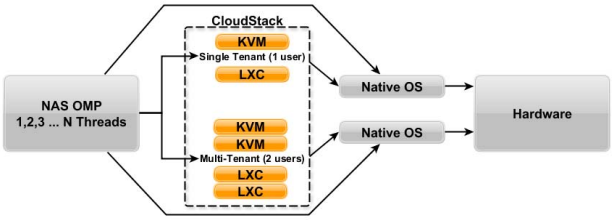


Fig. 1. High-level representation of the methodology followed in the evaluation.

class B[3], i.e., a medium sized incarnation of the benchmark, which means that the applications will assume: (I) 20^{30} random-number pairs for EP; (II) a grid with size of 512x256x256 and 20 iterations for FT; (III) IS will sort 2^{25} number of keys with maximum value of 2^{21}; (IV) a grid with size of 256x256x256 and 20 iterations for MG; (V) and CG will have 75000 number of rows with 13 values non-zero and 75 iterations to perform with an eigenvalue shift of 60 [14]. In addition, we create two main scenarios: The first one is a single tenant where CloudStack LXC-based cloud and CloudStack KVM-based cloud instances were deployed with full machine resources. The second is a multi-tenant scenario where two CloudStack instances of LXC-based cloud and two CloudStack instances of KVM-based cloud were deployed and the service offer its a half of the total host resources. The number of threads used were limited to the number of vCPUs available. For instance, in the single tenant scenario, we run NPB-OMP up to 8 threads and in the multi-tenant scenario we run NPB-OMP up to 4 threads. This was done to test multithreading applications, since we do not focus on over-commitment of resources. To create a baseline of results we also measured running on the native hardware, without virtualization. For instance, we define as scenarios the single and multi-tenants and environments as KVM-based cloud, LXC-based cloud and Native.

IV. BACKGROUND

This section presents an overview of the virtualization platforms (KVM and LXC) as well as the benchmark suite used in our evaluation (NPB).

A. Kernel-based Virtual Machine

KVM is an open source solution which creates a full virtualization environment and enable Linux x86 to create VMs with unmodified guest operating systems [15]. It utilizes virtualization extensions (Intel VT-x or AMD-V), which allows a user space program to utilize the hardware virtualization features of various processors [16]. Regarding to Linux process, KVM is treated as regular one through the *Quick Emulator* (QEMU) [13]. In addition, KVM does not perform emulation, instead it uses and provides the /dev/kvm/ device, which sets up the VM address space and feeds the simulated I/O trough QEMU [17].

[2]http://larcc.setrem.com.br/

[3]https://www.nas.nasa.gov/publications/npb_problem_sizes.html

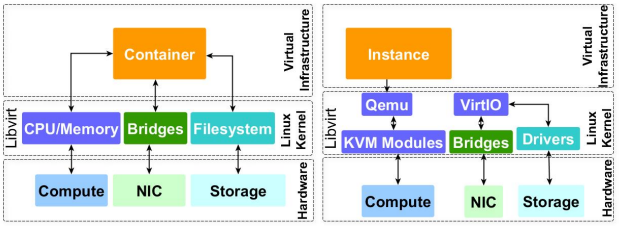


Fig. 2. LXC(Left) and KVM (Right) Overview [6].

B. Linux Containers

LXC is an OS-level virtualization, which abstracts computational resources through *Control Groups* (cgroups) to control system resources via namespaces, create and isolate the objects within the called container [15], [18]. These containers share the kernel with the host OS and thus, its processes and file system are accessible from the host. Therefore, it is defined as being a lightweight virtualization, which does not require physical hardware emulation. The main purpose of using Linux Containers is to provide resources or the same environment such as a VM. Finally, through a combination of kernel security features, it makes possible to create a virtual environment on the same machine without a hypervisor [15], [19].

Containers can divide resources that are managed by an operating system, into isolated groups to balance demands on resource usage. Furthermore, LXC can execute native instructions to the CPU core without any special interpretation mechanism. Thus, through the use of containers, the OS gives the applications the illusion of running on separate machines while sharing the underlying resources [20]. Figure 2 shows a comparison between LXC (OS-level virtualization) and KVM (full-virtualization). Both technologies use the Libvirt API. As can be seen LXC requires less software abstraction, using the same kernel of native OS and using Linux Bridges or native interfaces for network connectivity, while KVM uses a paravirtualized VirtIO driver for I/O operations and network connectivity offered to VMs.

In this paper, we aim at address performance aspects on private cloud deployments by using open source solutions. The cloud instances are deployed using KVM and LXC and are statistically compared and analysed.

C. NAS Benchmark Kernels

In order to assess the performance and feasibility when running real-world applications on cloud scenarios, we chose the NAS Parallel Benchmarks, which is a suite of applications designed to aid performance benchmarking of parallel supercomputers [21]. Derived from *computational fluid dynamics* (CFD) applications, these benchmarks consist of five kernels and three pseudo-applications including new benchmarks such as unstructured adaptive mesh, parallel I/O, multi-zone applications, and computational grids [22]. Each application has its particular characteristics, and these will be described in the evaluation. The benchmark set that we chose included integer sort, floating-point performance, calculation of matrix and memory intensive communication fields. For this evaluation, we consider only the five kernels from NPB suite which are described below.

Kernel IS: Integer Sort Sorts whole numbers using "bucket sort". IS Kernel performs a large integer sort operation, important in "particle method" codes. It makes tests both integer computation speed and communication performance. This benchmark is a specific generator of a large array by a scheme and then sort it [23].

Kernel EP: An Embarrassingly Parallel Benchmark Independent generation of Gaussian values and random variables using the Polar Marsaglia method. EP Kernel provides an estimate of the attainable limits for floating-point performance, that is, processor without interprocessor communication. EP kernel measures floating-point performance by tabulating statistics on pseudo-random data. It displays the simplest possible communication pattern between processes [24].

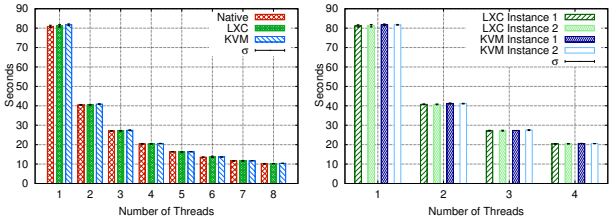
Kernel CG: Conjugate Gradiant Calculation of matrix values. This application uses a gradient method to compute an approximation to the smallest eigenvalue of a matrix. It is typical of unstructured grid computations, testing long and irregular communication, using unstructured matrix vector multiplication.

Kernel MG: Multi Grid Intensive communication for short and long-distance memory. It is a simplified multigrid calculation. It requires long and structured communication, short and long distance data communication test.

Kernel FT: Fast Fourier Transform Fourier transform method, using all-to-all communication. It is 3-D partial differential equation solved using FFTs, performing the essence of many "spectral" codes. As a definition, it is a rigorous test of heavy long-distance communication performance.

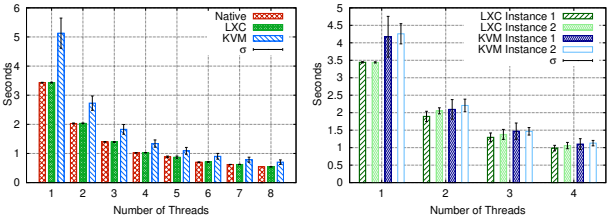
V. Performance Evaluation

The results obtained (execution time of three distinct environments (Native, LXC and KVM) and two scenarios (single and multi-tenants) are shown and described with a specific color in the graphs below. In the single tenant scenario, we identified the environments with red for the Native, green for LXC and blue for KVM. On the other hand, in the multi-tenant scenario, the environments were identified with dark-green for LXC Instance 1, light-green for LXC Instance 2, dark blue for KVM Instance 1 and light blue for KVM Instance 2 respectively. We plotted the execution time for each application in such a way that each thread has an available core/vCPU. In this approach, we consider cloud instances with a variant number of vCPUs and memory available. Also, the applications executed inside the instances used a different number of threads. We considered the total time (not only the parallel region) for all the applications. At first glance, we can observe that KVM-based cloud scenario presented the worst results, indicating that this kind of virtualization imposes a considerable overhead compared to LXC and Native environments .



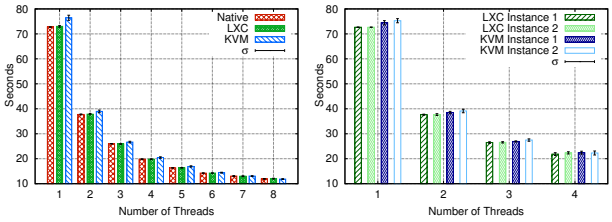
(a) Single-Tenant. (b) Multi-Tenant.

Fig. 3. EP Execution Times.



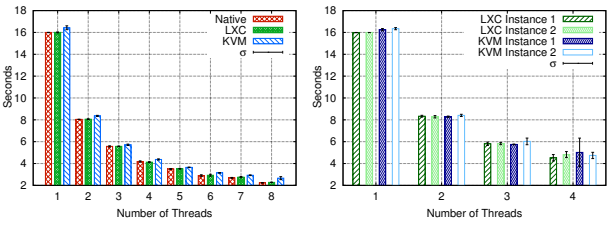
(a) Single. (b) Multi-Tenant.

Fig. 6. IS Execution Times.



(a) Single Tenant. (b) Multi-Tenant.

Fig. 4. FT Execution Times.

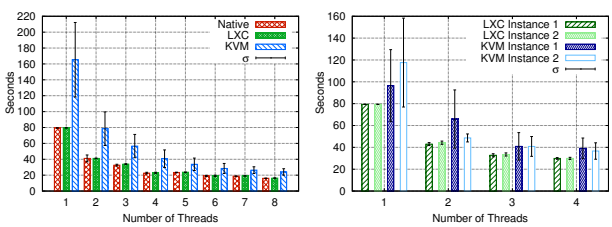


(a) Single Tenant. (b) Multi-Tenant.

Fig. 7. MG Execution Times.

EP kernel (Figure 3) has its operations totally independent in each thread number configuration. The operations involve the generations of Gaussian random variants. This application presents the actual context of parallel processing. It is noticed that in each increasing number of threads, the execution time has decreased considerably. This means that the performance gain follows at levels close to the recommended ones, with small differences between the environments. We observe that in both single (Figure 3(a)) and multi-tenants (Figure 3(b)), this kernel provides similar results in all environments (LXC-based cloud, KVM-based clouds and native). Moreover, the results obtained of Hashimoto, et al, shown that the performance degradation in EP is minimum because it requires less memory capacity [25].

FT (Figure 4) is a memory intensive application, focused on "everything for all communication". It presents some performance losses in the first thread at the full virtualized environment (KVM-based) with single tenant (Figure 4(a)). As this application uses a large amount of memory, the virtualization penalty is most significant. Moreover, we can observe that in the environment where are competition for resources between the tenants (Figure 4(b)), the difference between LXC-based cloud and KVM-based cloud is reduced. This behaviour is expected because KVM provides resource isolation [26].



(a) Single-Tenant. (b) Multi-Tenant.

Fig. 5. CG Execution Times.

Similarly, CG (Figure 5) and IS (Figure 6) presented a visual performance difference between the cloud scenarios in both single and multi-tenants. In CG application, there are some vector with vector multiplication operations and a vector with matrix multiplication operation, which generates a large volume of memory updates at the end of each loop iteration. As expected, the full virtuatization had worse results and a considerable overhead with respect to containers in single and multi-tenants. Moreover, KVM-based cloud provides results with a significant standard deviation. In the multi-tenant scenario (5(b)), we can see different results between the tenants. In addition, Regola, et al. emphasized that virtualization penalty is most significant in benchmarks that requires large amounts of communication or memory access [27]. Based on our results, we can also conclude that LXC-based cloud (single and multi-tenants) had low performance overhead when compared to the Native environment represented in microseconds.

In IS, the bucket sort algorithm performs operations at different memory positions to classify a vector of integer values. Also, this application has some specific characteristics, such as the smallest set of work and the fastest execution time among all applications in the NAS OpenMP parallel benchmark. Note that IS running in KVM (blue color) both single and multi-tenants performed even worse without using parallelism (Figure 6). In addition, this application provides a significant amount of standard deviation on KVM-based cloud. On multi-

tenant scenario (Figure 6(b)), the competition for resources make the tenants perform differently. IS scalability problem was already reported by Strazdins, et al. [7]. Moreover, we can note that LXC-based cloud has a near native performance with single tenants. However, with multi-tenants (Figure 6(b)), the difference between LXC-based cloud and KVM-based cloud decreases as the number of threads increases. Therefore, taking into account the standard deviation both have similar results with 4 threads.

MG application provides a solution to a three-dimensional discrete Poisson equation using Multi Grid approach. The impact of KVM overhead are smaller in relation to other methods as can be visualized in Figure 7. Although MG is a memory intensive kernel, the memory access seems not impact significantly. The single tenant (Figure 7(a)) LXC-based clouds had close results to the native environment. In multi-tenant scenario (Figure 7(b)), both LXC and KVM-based clouds provided similar results.

Our experiments led us to a deeper investigation about the technologies that can affect the obtained results. We identified that the *Kernel Samepage Mergin* (KSM) [28] increases memory density generating shared pages by merging equal ones. Consequently, there will be more memory available for running more VMs or applications on the same system [29]. In addition, both KVM-based cloud and LXC-based cloud can take advantage mainly in the multi-tenant environment on which the available memory is divided by two tenants. As these tenants are executing the same applications, KSM merge equals pages requisition. Particularly, this technology impacts directly in multi-tenant scenario of EP and MG applications, which uses the cache memory efficiently and presents close results between LXC and KVM-based clouds.

Moreover, Intel VT-x virtualization support in our machines may also impact on the performance. Neiger, et al. emphasize that this technology makes KVM run instructions with native access to CPU cores [30]. This technology impacts over the same applications which makes efficient use of cache memory (EP and MG). Moreover, the application characteristics combined with the experimental setup of our tests have resulted in an optimized performance.

VI. Statistical Analysis

We performed a statistical analysis to test the following hypotheses:

- $H_0 : LXC == KVM.$
  The performance when running a given NAS OpenMP parallel benchmark in a LXC-based cloud instance is equal when running it in a KVM-based cloud instance.
- $H_1 : LXC =! KVM.$
  The performance when running a given NAS OpenMP parallel benchmark in a LXC-based cloud instance is different when running it in a KVM-based cloud instance.

We used the IBM SPSS[4] to perform the statistical hypothesis tests. In this process, we first performed the sample collection. Subsequently, the normality test was done. This test is executed to determine if the sample distribution is a normal curve, determining whether the sample is parametric or non-parametric for the hypothesis test. The normality exists when the Kolmogorov-Smirnov and Shapiro-Wilk approaches have a significance greater than 0.05 (Sig > 0.05). Considering that the samples in this article are less than 31, the author [31] recommends the use of Shapiro-Wilk approach that is for small samples.

The next step was to perform the parametric or non-parametric test in each sample previously defined by the normality test. The T Test was applied in the parametric samples, which paired the samples and compared them to obtain the result of significance. On the other hand, the Wilcoxon Test was applied in the non-parametric samples, which ranks the samples and also compares them to obtain the significance. The significance of these tests (T test and Wilcoxon Test) will indicate whether the comparison is significantly different or not using the 95% confidence level. The use of this statistical process allows small differences to be detected according to [31].

Table II and III presents the results of the comparison between LXC and KVM with sigle and multi-tenants respective as well as the hypothesis test for H_0 and H_1. In the columns "LXC-OMP α" and "KVM-OMP ω" lines with gray background indicate non-parametric test and lines with white background indicate parametric test performed. Also, in the "$Sig.$" column numbers with gray background highlights that LXC had the best result in this specific hypothesis test. The numbers with black background and white font highlight that KVM had the best execution time in this specific hypothesis test. The numbers with white background and black font we accept H_0, which means that the difference is not significant between LXC and KVM environments. Note that in Table II (single tenant) almost all the tests we rejected H_0 and accepted H_1.

Observing the arithmetic means, we identified that LXC-based cloud instances presented better results in the majority of the tests (93.75%) with significant differences. The KVM presented better results only in the Kernel FT with eight threads (1.56%). Finally, in 4.69% of the tests, it is not possible to say that there are significant differences. In addition, Table III (multi-tenant) shows that in 45% of the tests, we can reject H_0 (no significant differences between the samples). The arithmetic means show that in 55% of the tests there are significant differences, and those highlighting results to LXC-based cloud.

VII. Conclusion

This paper presented a performance analysis and evaluation, using a statistical approach to identify if the results are significantly different. Our work covered experiments performed in two scenarios (single and multi-tenants) over private cloud conditions deployed with the CloudStack platform. The environments supported both KVM and LXC virtualization technologies, where the NAS OpenMP parallel benchmarks were

[4]https://www.ibm.com/us-en/marketplace/spss-statistics

TABLE II
STATISTICAL ANALYSIS - LXC VS KVM - SINGLE TENANT.

		LXC-OMP (α)		KVM-OMP (ω)		α VS ω
Bench	Th	$\bar{x}$ (10x)	σ (10x)	$\bar{x}$ (10x)	σ (10x)	$Sig.$
IS	1	3,434	0,014	5,131	0,545	0,005
	2	2,035	0,014	2,729	0,260	0,000
	3	1,394	0,015	1,83	0,175	0,000
	4	1,027	0,007	1,341	0,131	0,000
	5	0,873	0,040	1,092	0,122	0,013
	6	0,709	0,007	0,904	0,103	0,005
	7	0,623	0,007	0,787	0,090	0,005
	8	0,542	0,004	0,703	0,081	0,005
EP	1	81,211	0,647	81,724	0,485	0,202
	2	40,461	0,092	40,905	0,155	0,005
	3	27,151	0,200	27,393	0,237	0,009
	4	20,36	0,138	20,551	0,133	0,025
	5	16,255	0,018	16,365	0,036	0,005
	6	13,704	0,382	13,648	0,021	0,720
	7	11,609	0,075	11,71	0,044	0,018
	8	10,188	0,081	10,378	0,082	0,012
CG	1	79,506	0,162	165,296	49,410	0,005
	2	41,095	0,368	78,467	22,265	0,005
	3	33,954	0,485	56,603	15,430	0,009
	4	23,043	0,713	40,707	11,671	0,005
	5	23,735	0,158	33,686	8,258	0,009
	6	19,196	0,803	28,437	6,663	0,008
	7	19,015	0,142	26,336	4,457	0,005
	8	16,332	0,297	24,019	4,270	0,005
MG	1	16,004	0,034	16,429	0,186	0,005
	2	8,061	0,032	8,361	0,053	0,000
	3	5,572	0,022	5,714	0,068	0,000
	4	4,129	0,041	4,372	0,056	0,005
	5	3,518	0,031	3,648	0,025	0,000
	6	2,915	0,087	3,147	0,023	0,005
	7	2,749	0,054	2,925	0,344	0,000
	8	2,278	0,020	2,663	0,146	0,005
FT	1	73,013	0,325	76,448	1,076	0,000
	2	37,855	0,247	38,964	0,547	0,001
	3	25,903	0,097	26,666	0,335	0,000
	4	19,807	0,180	20,469	0,315	0,005
	5	16,35	0,085	16,9	0,274	0,000
	6	14,269	0,077	14,422	0,178	0,034
	7	13,014	0,164	13,055	0,178	0,602
	8	12,015	0,176	11,855	0,164	0,022

TABLE III
STATISTICAL ANALYSIS - LXC VS KVM - 2 MULTI-TENANTS.

		LXC-OMP (α)		KVM-OMP (ω)		α VS ω
Bench-UserN	Th	$\bar{x}$ (10x)	σ (10x)	$\bar{x}$ (10x)	σ (10x)	$Sig.$
IS-USER1	1	3,443	0,018	4,173	0,614	0,005
	2	1,895	0,153	2,096	0,294	0,153
	3	1,296	0,130	1,472	0,245	0,103
	4	0,988	0,081	1,103	0,162	0,059
IS-USER2	1	3,439	0,017	4,258	0,304	0,000
	2	2,054	0,091	2,207	0,190	0,077
	3	1,379	0,150	1,468	0,112	0,235
	4	1,06	0,095	1,130	0,081	0,061
EP - USER1	1	81,117	0,633	81,705	0,490	0,037
	2	40,755	0,247	41,192	0,322	0,002
	3	27,18	0,222	27,298	0,020	0,308
	4	20,39	0,155	20,568	0,113	0,053
EP-USER 2	1	81,221	0,697	81,728	0,193	0,052
	2	40,722	0,227	41,129	0,169	0,009
	3	27,214	0,230	27,499	0,253	0,022
	4	20,451	0,160	20,530	0,054	0,150
CG-USER1	1	79,363	0,143	96,507	34,734	0,005
	2	42,987	1,425	65,931	28,072	0,022
	3	32,769	1,470	40,916	13,249	0,047
	4	29,849	0,819	39,089	9,957	0,022
CG-USER2	1	79,382	0,158	117,611	42,876	0,005
	2	44,258	1,546	48,639	3,854	0,003
	3	33,482	1,523	40,834	9,560	0,022
	4	29,9	1,030	36,656	7,923	0,028
MG-USER1	1	15,99	0,009	16,258	0,106	0,005
	2	8,332	0,084	8,288	0,060	0,129
	3	5,809	0,158	5,748	0,033	0,263
	4	4,527	0,312	5,020	1,374	0,221
MG-USER2	1	15,992	0,007	16,347	0,010	0,000
	2	8,281	0,113	8,405	0,105	0,088
	3	5,822	0,098	6,026	0,322	0,036
	4	4,82	0,289	4,740	0,303	0,475
FT-USER1	1	72,679	0,096	74,458	0,902	0,005
	2	37,651	0,214	38,481	0,506	0,001
	3	26,467	0,325	26,914	0,248	0,000
	4	21,864	0,585	22,430	0,601	0,093
FT-USER2	1	72,68	0,077	75,281	0,837	0,005
	2	37,677	0,866	39,119	0,739	0,001
	3	26,562	0,283	27,480	0,493	0,001
	4	22,353	0,465	22,298	0,837	0,575

experimented. We concluded that LXC provides the smaller overhead for this kind of applications. In the first scenario when comparing the instance types, container-based virtualizations outperforms kernel-based virtualization on 93.75% under private cloud conditions with a CloudStack platform for our samples. Only for an exceptional case (FT with eight threads) that KVM-based instance outperform LXC with minimal difference. Moreover, LXC-based cloud outperforms KVM on 55% of the results. In 45% there are no significant differences. Note that the percentage is referred to the number of "$Sig.$" that indicates significantly different with better execution time for each scenario. Therefore, we can highlight that with multi-tenants, the KVM-based cloud had results closer to the LXC-based clouds.

We also discovered that the high memory usage and access in these applications impacted significantly the performance of KVM-based instances. That is because full virtualization adds more instructions that need to be managed by the CPU. Consequently, it has to treat more information and bufferization that causes performance degradation compared with the native environment performance. Moreover, KSM and Intel VT-x are permissive technologies for better performance, which can impact significantly on applications which performs efficient cache memory usage.

Finaly, LXC-based cloud is the best option for scientific applications with single tenants in our tests. Despite that LXC-based cloud had also better results with multi-tenants, the number of results that there is no significant differences between the cloud environments is about 45%. Although multi-tenants

LXC-based cloud was the best option, KVM-based cloud can still be considered a suitable option. The performance increasing of KVM-based cloud with multi-tenants is due to the isolation of resources and users. On the other hand, LXC has the best results with single tenants and a worse resource isolation.

In the future, we plan to evaluate different application domains (*e.g.*, Stream Processing, Big Data, Deep Learning, and Machine Learning) to discover different behaviours; include different IaaS management tools (*e.g.*, OpenStack, OpenNebula); perform the experiments with overprovision of computer resources or even multi-tenancy, which are practices widely used in the Cloud; and perform our experiments with other virtualization technologies (*e.g.*, HyperV, VMWare, Xen, Docker).

ACKNOWLEDGMENT

The authors would like to thank for the partial support from HiPerfCloud project, CAPES, FAPERGS, and the institutional support of SETREM, PUCRS, UNIPAMPA, and FAU.

REFERENCES

[1] V. Chang, R. J. Walters, and G. Wills, "The Development that Leads to the Cloud Computing Business Framework," *International Journal of Information Management*, vol. 33, no. 3, pp. 524 – 538, 2013.

[2] A. S. Tanenbaum and T. Austin, *Organização Estruturada de Computadores*. Pearson, 2013.

[3] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema, "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 6, pp. 931–945, June 2011.

[4] A. Vogel, D. Griebler, C. A. F. Maron, C. Schepke, and L. G. Fernandes, "Private IaaS Clouds: A Comparative Analysis of OpenNebula, CloudStack and OpenStack," in *24th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*. Heraklion Crete, Greece: IEEE, Febuary 2016, pp. 672–679.

[5] E. Walker, "Benchmarking amazon ec2 for hig-performance scientific computing," *; login:: the magazine of USENIX & SAGE*, vol. 33, no. 5, pp. 18–23, 2008.

[6] A. Vogel, D. Griebler, C. Schepke, and L. G. Fernandes, "An Intra-Cloud Networking Performance Evaluation on CloudStack Environment," in *25th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*. St. Petersburg, Russia: IEEE, March 2017, p. 5.

[7] P. E. Strazdins, J. Cai, M. Atif, and J. Antony, "Scientific application performance on hpc, private and public cloud resources: A case study using climate, cardiac model codes and the npb benchmark suite," in *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*. IEEE, 2012, pp. 1416–1424.

[8] D. Griebler, A. Vogel, C. A. F. Maron, A. M. Maliszewski, C. Schepke, and L. G. Fernandes, "Performance of data mining, media, and financial applications under private cloud conditions," in *23rd IEEE Symposium on Computers and Communications (ISCC)*. Natal, Brazil: IEEE, 2018.

[9] E. Roloff, M. Diener, A. Carissimi, and P. O. Navaux, "High Performance Computing in the cloud: Deployment, performance and cost efficiency," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*. IEEE, 2012, pp. 371–378.

[10] T. K. Okada, A. Goldman, and G. G. H. Cavalheiro, "Using NAS Parallel Benchmarks to Evaluate HPC Performance in Clouds," *International Symposium on Network Computing and Applications*, pp. 1–4, 2016.

[11] 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.

[12] M. Kang, D. I. Kang, J. P. Walters, and S. P. Crago, "A comparison of system performance on a private openstack cloud and amazon ec2," in *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, June 2017, pp. 310–317.

[13] J. Zhang, X. Lu, and D. K. Panda, "Performance characterization of hypervisor-and container-based virtualization for hpc on sr-iov enabled infiniband clusters," in *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, May 2016, pp. 1777–1784.

[14] D. Griebler, J. Löff, L. Fernandes, G. Mencagli, and M. Danelutto, "Efficient nas benchmark kernels with c++ parallel programming," in *26 Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, 01 2018.

[15] LXC, "Linux containers <https://linuxcontainers.org/lxc/introduction/>," 2018, last access mar, 2018.

[16] KVM, "Kernel virtual machine <https://www.linux-kvm.org/page/main_page>," 2018, last access mar, 2018.

[17] B. Zhang, X. Wang, R. Lai, L. Yang, Z. Wang, Y. Luo, and X. Li, "Evaluating and optimizing i/o virtualization in kernel-based virtual machine (kvm)," in *Proceedings of the 2010 IFIP International Conference on Network and Parallel Computing*, ser. NPC'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 220–231. [Online]. Available: http://dl.acm.org/citation.cfm?id=1882011.1882036

[18] R. Dua, A. R. Raja, and D. Kakadia, "Virtualization vs containerization to support paas," in *2014 IEEE International Conference on Cloud Engineering*, March 2014, pp. 610–614.

[19] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 3, pp. 275–287, Mar. 2007. [Online]. Available: http://doi.acm.org/10.1145/1272998.1273025

[20] M. Helsley, "Lxc: Linux container tools <https://www.ibm.com/developerworks/linux/library/l-lxc-containers/>," 2009, last access mar, 2018.

[21] D. H. Bailey, "Nas parallel benchmarks," in *Encyclopedia of Parallel Computing*. Springer, 2011, pp. 1254–1259.

[22] NPB, "Npb <https://www.nas.nasa.gov/publications/npb.html>," 2018, last access mar, 2018.

[23] D. H. Bailey, "The nas parallel benchmarks," in *Encyclopedia of Parallel Computing*. Springer, November 2009.

[24] J. Fernández, M. Anguita, E. Ros, and J. L. Bernier, "Sce toolboxes for the development of high-level parallel applications," in *Computational Science – ICCS 2006*, V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 518–525.

[25] Y. Hashimoto and K. Aida, "Evaluation of performance degradation in hpc applications with vm consolidation," *Third International Conference on Networking and Computing*, pp. 1–5, 2012.

[26] A. A. A. Mardan and K. Kono, "Containers or hypervisors: Which is better for database consolidation?" in *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, Dec 2016, pp. 564–571.

[27] N. Regola and J. C. Ducom, "Recommendations for virtualization technologies in high performance computing," in *2010 IEEE 2º CloudCom*, Nov 2010, pp. 409–416.

[28] KSM, "Kernel samepage merging <https://www.linux-kvm.org/page/ksm>," last acess mar, 2018.

[29] A. Arcangeli, I. Eidus, and C. Wright, "Increasing Memory Density by Using KSM," in *Proceedings of the Linux Symposium*, 2009, pp. 19–28.

[30] G. Neiger, A. Santoni, F. Leung, D. Rodgers, and R. Uhlig, "Intel Virtualization Technology: Hardware Support for Efficient Processor Virtualization." *Intel Technology Journal*, vol. 10, no. 3, 2006.

[31] A. Field, *Discovering Statistics Using IBM SPSS Statistics*, ser. Discovering Statistics Using IBM SPSS Statistics: And Sex and Drugs and Rock 'n' Roll. SAGE Publications, 2013.