applied sciences

MDPI

Article

Benchmarking and Performance Evaluations on Various Configurations of Virtual Machine and Containers for Cloud-Based Scientific Workloads

Syed Asif Raza Shah [1], Ahmad Waqas [1], Moon-Hyun Kim [2], Tae-Hyung Kim [3], Heejun Yoon [4] and Seo-Young Noh [2,*]

[1] Department of Computer Science and CRAIB, Sukkur IBA University (SIBAU), Sukkur 65200, Pakistan; asif.shah@iba-suk.edu.pk (S.A.R.S.); ahmad.waqas@iba-suk.edu.pk (A.W.)
[2] Department of Computer Science, Chungbuk National University, Cheongju-si 28644, Korea; moonhyun.dev@gmail.com
[3] Samsung Electronics, Seoul 135856, Korea; thkim4u@gmail.com
[4] Global Science Experimental Data Hub Center, Korea Institute of Science and Technology Information, 245 Daehak-ro, Yuseong-gu, Daejeon 34141, Korea; k2@kisti.re.kr
* Correspondence: rsyoung@cbnu.ac.kr

Abstract: Cloud computing manages system resources such as processing, storage, and networking by providing users with multiple virtual machines (VMs) as needed. It is one of the rapidly growing fields that come with huge computational power for scientific workloads. Currently, the scientific community is ready to work over the cloud as it is considered as a resource-rich paradigm. The traditional way of executing scientific workloads on cloud computing is by using virtual machines. However, the latest emerging concept of containerization is growing more rapidly and gained popularity because of its unique features. Containers are treated as lightweight as compared to virtual machines in cloud computing. In this regard, a few VMs/containers-associated problems of performance and throughput are encountered because of middleware technologies such as virtualization or containerization. In this paper, we introduce the configurations of VMs and containers for cloud-based scientific workloads in order to utilize the technologies to solve scientific problems and handle their workloads. This paper also tackles throughput and efficiency problems related to VMs and containers in the cloud environment and explores efficient resource provisioning by combining four unique methods: hyperthreading (HT), vCPU cores selection, vCPU affinity, and isolation of vCPUs. The HEPSCPEC06 benchmark suite is used to evaluate the throughput and efficiency of VMs and containers. The proposed solution is to implement four basic techniques to reduce the effect of virtualization and containerization. Additionally, these techniques are used to make virtual machines and containers more effective and powerful for scientific workloads. The results show that allowing hyperthreading, isolation of CPU cores, proper numbering, and allocation of vCPU cores can improve the throughput and performance of virtual machines and containers.

Keywords: cloud computing; virtual machines; containers; performance; throughput; virtualization; isolation

Citation: Shah, S.A.R.; Waqas, A.; Kim, M.-H.; Kim, T.-H.; Yoon, H.; Noh, S.-Y. Benchmarking and Performance Evaluations on Various Configurations of Virtual Machine and Containers for Cloud-Based Scientific Workloads. *Appl. Sci.* **2021**, *11*, 993. https://doi.org/10.3390/app11030993

Academic Editor: Fabrizio Marozzo
Received: 16 December 2020
Accepted: 20 January 2021
Published: 22 January 2021

1. Introduction

Recently, cloud computing [1] has become the most promising computing paradigm that provides flexible and on-demand infrastructure to scientific workloads. It has evolved from grid and utility computing. Being emerged from these computing paradigms, cloud computing is recently considered as an alternative to grid, cluster, and supercomputing for scientific workloads [2] because of the characteristics of cloud computing such as scalability, on-demand self-service, elasticity, and availability. In cloud environments, users do not need to worry about system implementation and administration, cloud computing becomes a desired tool that works as infrastructure-as-a-service (IaaS) and

fulfills the necessity of computing resources [3]. Scientific workloads manipulated using high performance computing (HPC), high throughput computing (HTC), and many-task computing (MTC) [4] can be executed in virtualized computing environment.

High performance tasks require a huge amount of computing power for a short period of time. In contrast, high throughput computing involves an enormous amount of computing power over a longer period of time, such as months or years. Multitask computing acts as a consensus solution to bridging the gap between HTC and HPC. It can perform many independent and dependent tasks using huge computing in shorter time. In MTC, task-parallel applications are performed on large-scale distributed systems. The major concerns related to scientific workloads are higher throughput and enhanced performance of virtualization or containerization in a cloud environment. In order to address these concerns, some techniques need to be proposed to improve the overall performance.

The Virtual Machine Monitor (VMM) or hypervisor [5] is a software abstraction layer that was introduced by virtualization technology. Cloud computing that uses virtual machines (VMs) for enabling a complete system with resource virtualization becomes most popular among other technologies. It makes physical infrastructures easy to manage and virtualizes full software stacks effectively with its operating system [6]. The VM is a computer system mirroring that provides real machines with functionality. It is regarded as the cloud environment's basic logical tool that provides computing facilities. VMM is an abstraction layer of the physical hardware and tracks virtual machines. It works with physical resources and logical resources. In addition, it also provides a complete view of heterogeneous underlying hardware that allows VMs to run on any computing system without considering the dependencies between software and hardware.

On the other hand, today's cloud service providers are also offering container deployment (e.g., Docker, LXC, etc.), which is becoming more popular than the VMs. The concept of container is similar to VM, but it consumes comparatively less time and resources. It is considered more as an application-specific solution in cloud environments. In containerization, the same kernel is being shared for containers and the host operating system; that is the key enabling feature of containers that make it lightweight as compared to VMs. In containerization, the hardware and software components are being shared between host the operating system (OS) and containers' applications. The host OS is mainly responsible for ensuring the isolation among the applications of containers. Because of single host OS, containers help to reduce the overhead of management as well.

Performance of non-virtualized environment differs from virtualized environment because of the interactions of virtual machines with the abstraction layer called VMM. Comparing the container's performance with bare metal is also different because of shared kernel. The main important factor for optimizing the VMs/containers is the efficiency and availability for scientific workloads. Many scientific tasks require successful preparation and fast execution to achieve useful scientific results. In order to obtain advantages of cloud computing, the issues related to efficiency and throughput need to be addressed directly in virtualized and containerized scientific cloud environments.

With the goal of addressing aforementioned challenges, this article proposes a method for solving these issues—performance and throughput. Currently, the scientific community is ready to work over the cloud as it is considered as a resource-rich paradigm. Cloud computing enables users to work anywhere by providing logical resources such as virtual machines or containers. However, it should be noted that there are a few VM/container associated issues regarding performance and throughput. In this paper, in order to utilize virtualization technologies, we evaluated the different configurations of VMs and containers, which are the main computing actors for scientific workloads. We also take into consideration the problems of throughput and efficiency related to VMs and containers, and explore efficient resource provisioning by combining four unique methods: hyper-threading (HT), vCPU cores selection, vCPU affinity, and isolation of vCPUs. The scope of this research is mainly focuses on scientific workloads. Furthermore, a balanced view of performance and throughput is also given. A renowned cloud computing platform,

OpenStack [7], has been adopted to configure the computing environment for logical setup and to run scientific applications. The HEPSPEC06 benchmark that is produced by the HEPiX CPU Benchmark Group [8] is used for performance evaluation of virtual machines and containers. Realistic issues regarding the performance of VMs/containers and throughput degradation are also investigated. In this paper, we use the combination of four famous techniques to achieve real-time performance and higher throughput of virtual machines and containers. These techniques are: hyperthreading technology, the selection of vCPU cores per virtual machine/container, physical CPU isolation, and pinning of vCPU cores of a virtual machine/container in a multicore NUMA (nonuniform memory access) architecture [9,10]. Our proposed solution shows that a fine combination can work very efficiently to achieve higher performance and enhanced throughput of VMs and containers in a cloud-based environment for scientific computing.

The rest of the paper is organized as follows. Section 2 provides an overview of related work. In Section 3, the HEPSPEC06 benchmark suite is discussed. Section 4 presents the proposed solution. Section 5 discusses the evaluation and results. Finally, Section 6 draws conclusions.

2. Related Work

Currently, many studies have been actively conducted by many researchers, addressing the performance evaluation of virtual machines. In literature, many articles work on increased performance of virtual machines. Most of the literature have used VMM abstraction layer in general to measure the overhead of VMs. In the article [10], the authors reported the degradation of performance in virtual machines. They worked on NUMA architectures, based on the effects of virtualization and tested the architecture with the hypervisors such as KVM and Xen. In [11], the authors measured one of the performance factors, i.e., startup-time of virtual machines in cloud environments. Those time measurements are performed and analyzed among three well-known cloud providers: Amazon EC2, Rackspace, and Microsoft Azure. The authors presented a systematic detailed analysis of cloud computing application efficiency evaluations for scientific workloads [2]. The result said that reliability and cloud efficiency are inadequate for scientific workloads. Infrastructure-as-a-service is the main feature for adopting cloud to cope with scientific workload. In [12], the authors presented a survey on performance overhead of virtual machines. They also discussed how the performance varies from single server virtualization to geo-distributed datacenters. The authors in [13] illustrated the impact of type of workload, processor pinning, configuration, and partial background CPU load. They also investigated and addressed the issues of paired colocated compute-intensive workloads that create interference and reduce the overall performance. OpenStack cloud platform for IaaS implementation evaluates energy efficiency in high-performance computing [14]. In this article, performance impact was evaluated, which is produced by the underlying hypervisors and the IaaS solution by using the HPCC (High-Performance Computing Cluster) and HPL (High-Performance Linpack) [15], and Graph500 benchmarks [16]. The author in [17] evaluated performance of virtual machines and organization of scientific workflow on virtual systems. They conducted a complete benchmark to test the CMS (Compact Muon Solenoid) scientific workloads on virtual machines. The quantitative analysis of the efficiency of Amazon EC2 (Public Cloud) for the workloads of HPC using benchmarks was addressed in [18].

In [19], performance of virtual machines in the cloud is compared with containers usage in the cloud. Virtual machines and Linux containers are compared in terms of network performance and reduction of potential performance overhead. Some literature work quantifies the overhead efficiency of VMs by using a VMM or Xen virtualization layer. To improve the resource utilization in cloud computing environments, swarm optimization-based workload optimization (SOWO) technique is proposed in [20]. In the research, they also used the OpenStack platform and claimed that resource utilization is increased by 50 percent in cloud computing environments. Operating containers on top of virtual

machines technique was examined in [21]. In the paper, they followed an observational approach to measure the overhead efficiency provided by the additional virtualization layer in virtual machines by performing different benchmarking tests and implementing programs with the real-world-use case scenarios. In [22], the authors compare the performance of a typical KVM hypervisor to a Docker Linux container by contrasting the performance of VMware Server with the actual physical servers. The extended paravirtualization (XPV) is another approach evaluated for effective virtualization of NUMA architecture [23]. XPV consists of revisiting the interface between the hypervisor and the host OS, and between the host OS and the device runtime libraries (SRLs), so that they can automatically take into account changes in the NUMA topology. The authors in [24] proposed an empirical overview of the success impact of the different resource affinities. They proposed a performance prediction model called resource affinity function effect estimation (RAIE). The RAIE model takes into account the real effect of resource affinity dependent on VM activities that can be tracked online without VM alteration. The proposed model tried to increase the average prediction accuracy of VMs.

It is difficult to find a study that analyzes the latest techniques for enhancing virtual machine efficiency and throughput. The major purpose of our research is to enhance overall efficiency and throughput of VMs and containers, so that scientific workloads can run more efficiently. The series of four well-known techniques is used to reduce the impact of results on virtual machines and containers.

3. Proposed Solution

This study proposes an approach based on four proven techniques in order to increase the overall performance of virtualization/containerization technologies and enable scientific communities that use the cloud computing environment for their workloads. These techniques are able to increase the overall efficiency of virtual machines for a scientific environment.

3.1. Overview

We enable the Intel HT (hyperthreading) technology to improve the performance of the CPU cores [25]. HT technology was the origin of bringing parallel computation to PCs. Using this technology, a single processor presented physically appears as two logical processors that share physical resources and use the duplicated architecture in the operating system. In other words, the operating system considers each CPU core as two separated CPUs. Hyperthreading technology improves utilization of CPU resources; if we enable the HT then we can utilize the resource efficiently. The performance of threaded applications is improved and processing throughput is also increased. Not only is HT used to maximize the performance of processors, but utilization of resources is enhanced that leads to the higher throughput for specific types of scientific workloads. The processing cores performance increases at some extent to their capacity by enabling HT on the system, where virtual machines and containers are running the scientific workloads. The first technique of study mentions that HT must be enabled on physical machines so that virtual machines or containers running scientific workload produces higher throughput. The underutilization of processor resources is also incorporated by HT technology. Application level performance and system level performance can be boosted using HT. The second proposed technique is used to improve performance of VMs or containers by selecting a greater number of vCPU (virtual CPU) cores that run on virtual machines or containers. Bare metal performs comparatively better than virtual machines and containers with higher vCPU cores in scientific workloads. This performance degradation is caused by the scheduling policy of hypervisor, and performance also decreases when loosely coupled scientific tasks with a higher number of vCPUs runs on virtual machines or containers. In VMs or containers, the performance of a single core might be increased when HT is enabled because it adds the additional vCPU cores to the virtual machine or container. This study proposes that to achieve real-time performance, virtual machines or containers

would operate with fewer vCPU core numbers. The third technique suggested is used to enhance the overall throughput and efficiency of virtual machines or containers by physically isolating the processor's cores. In the host operating system, there are several interferences running on the physical processing core, which decrease virtual machine or container performance. Complete isolation of physical processing cores from the kernel of the host OS improves the performance of VMs or containers. Using isolation, tasks in the system do not follow scheduler assignment. The fourth technique suggests pinning of CPU cores of VM or containers. For scientific workloads, CPU cores must be pinned properly in order to improve performance. This pinning is required to consider NUMA multicore processor architecture. Pinning needs to be performed on every logical CPU of the guest virtual machine or container using the ID of host system. To execute the scientific workloads efficiently, the NUMA architecture with pinning technique can maximize the overall performance of virtual machines and containers. The proposed combinations of four strategies are evaluated with several modifications to achieve higher efficiency and performance of the virtual machine and container in the cloud computing world.

3.2. Experimental Setup

The operational private cloud environment is used for this experimental setup to test the efficiency and output of virtual machines and containers. For our all experiments, we utilized the OpenStack cloud operating system, which is an open source cloud management solution. In this experiment, a cloud environment was set with the composition of fifteen physical machines. Table 1 depicts the specification of every physical machine.

Table 1. Specification of the physical machine.

Description	Specs
Architecture	x86_64
Processor Type	Intel Xeon 2.6 GHz CPU
Total Number Cores	16 (without HT)/32 (with HT)
Total RAM	96 GB
Storage Capacity (HDD)	350 GB
NUMA Nodes	2 (Node 0 = 8 cores and Node 1 = 8 cores)
NICs	3 × 1 Gbps Ethernet
Operating System	CentOS 7.0
Hypervisor	KVM
Linux Kernel	v3.10
Compiler and flags	Compiler: gcc-4.8, Flags: -O2-pthread-fPIC-m32

The aim of this benchmarking procedure is to measure the CPU performance of a particular worker node in our cloud computing environment for scientific workloads. According to HEPiX recommendation, we used the recommended version of OS and kernel (CentOS 7 or Scientific Linux 7 and Kernel v3.1) to achieve better performance. In our experimental setup, we used GNU Compiler Collection (gcc-4.8) default compiler along with following compiler flags: -O2-pthread-fPIC-m32.

Figure 1 shows complete connectivity and flow of the experimental setup with the fifteen machines. OpenStack is the free cloud platform with several nodes for providing computing resources and other services. The system is composed of network, controller, storage, and several compute nodes. The central management is provided by the controller node in this cloud setup environment. The storage node provides capacity to store images and instances and the network node provides the services related to communication and networking in this environment. This experimental setup consists of twelve compute nodes for providing resources to VMs or containers. These nodes play an essential role for our all experiments because each virtual machine or container uses their computing resources in order to work as a physical machine for scientific jobs.

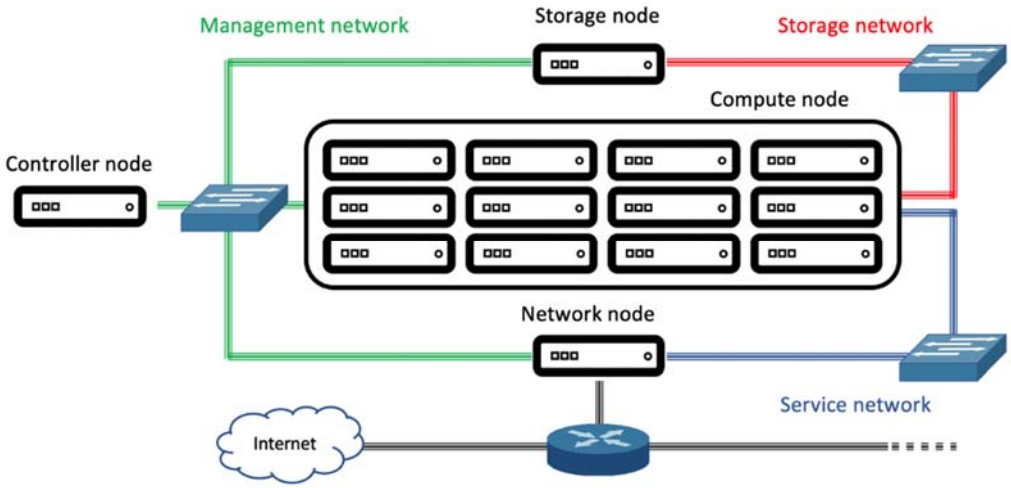

Figure 1. OpenStack-based experimental setup and connectivity.

4. Evaluation and Test Cases

HEPSPEC06 is based on a benchmark specific high-energy physics (HEP) suite derived from the SPEC06 CPU benchmark [26]. This benchmark suite delivers better performance, as it uses 64-bit mode mainly. The HEPSPEC06 benchmark is used to test CPU core performance. Before this benchmark set, SPECint 2000 [26] was regarded as the most common benchmark in the high-energy physics community, withdrawn in February 2007. This benchmark was introduced in the period of 2007–2008 by HEPiX CPU Benchmark working group [8]. HEPiX group puts efforts together to launch this benchmark that is considered as the replacement of SPECint 2000. It replaced the outdated kSI2k metric. Briefly, this benchmark is used to measure a CPU's performance. It is suitable for the operations required for floating point calculations.

The HEPSPEC06 test performs different tests on a single machine for each core at the same time. Since this test can perform integer operations and floating point closely linked to HEP codes, the scientific communities of physics and IT disciplines would likely use this benchmark. The HEPSPEC06 test also provides versatility on both 32-bit and 64-bit architecture to run the benchmarks. For optimizing CPU performance of virtual machines and containers in the cloud environment, HEPSPEC06 is the ideal benchmark. In this experimental setup, the HEPSPEC06 test suite is used to evaluate performance of VMs and containers supplied for scientific workload execution.

As noted, the HEPSPEC06 benchmark is used to test the efficiency and performance of virtual machines and containers, so that high-power scientific workloads can be carried out on the experimental setup presented above. Throughout this evaluation process, the HEPSPEC06 benchmark runs on VMs or containers to analyze better performance and throughput as well. In this experiment, normal CLI (command line interface) commands on OpenStack are used for provisioning virtual machines and containers. Each virtual machine is initialized and controlled by an independently uploaded image that is installed with the basic operating system of CentOS 7.1, which works the same as the host operating system. On the other hand, the setup of containers also performed similarly in the cloud. Required necessary packages are also installed to run the HEPSPEC06 benchmark in virtual machines and containers. Three different test cases are evaluated in conjunction with techniques to increase VMs and containers' efficiency and throughput. In this section, we discuss the detailed results of all test cases.

4.1. Test-1

Our first experiment of test-1 depends on the size of the VM and container by selecting the vCPU cores per virtual machine or container. In the cloud environment, hyperthreading remains disabled on the physical host. To set up baseline results to equate with virtual

machine or container performance, the HEPSPEC06 test is executed on a cloud physical host. In our current setup, five separate configurations are executed on the physical host system, depending on the selection of vCPU cores for a single VM or container. Table 2 shows evaluation of each configuration individually. The HEP-SPEC06 benchmark is running simultaneously on all virtual machines or containers in single configuration. With each virtual machine or container in the described setup, we assigned 20 GB of local permanent storage for data. Regarding RAM, it is divided equally among VMs or containers. For example, a single physical machine has a maximum of 96 GB RAM, and we are running 8 VMs/containers on it; this means each VM/container will get 12GB RAM. We also added the RAM information into Tables 2 and 3.

Table 2. Five different configurations of VMs/containers for test-1.

Configurations	Number of Containers	Number of VMs	Number of vCPUs Per VM/Container	Per VM/Container RAM Allocation
Configuration #1	1 Container	1 VM	16 vCPUs	96 GB
Configuration #2	2 Containers	2 VMs	8 vCPUs	48 GB
Configuration #3	4 Containers	4 VMs	4 vCPUs	24 GB
Configuration #4	8 Containers	8 VMs	2 vCPUs	12 GB
Configuration #5	16 Containers	16 VMs	1 vCPU	6 GB

Table 3. Five different configurations of VMs/containers for test-2.

Configurations	Number of Containers	Number of VMs	Number of vCPUs Per VM/Container	Per VM/Container RAM Allocation
Configuration #1	1 Container	1 VM	32 vCPUs	96 GB
Configuration #2	2 Containers	2 VMs	16 vCPUs	48 GB
Configuration #3	4 Containers	4 VMs	8 vCPUs	24 GB
Configuration #4	8 Containers	8 VMs	4 vCPUs	12 GB
Configuration #5	16 Containers	16 VMs	2 vCPUs	6 GB

4.2. Test-2

In the second test case, a separate (compute node) physical host of the cloud is used. Hyperthreading is enabled in this case with five different virtual machine and container configurations, as shown in Table 3. The configurations may seem identical with the previous case, but the difference is that each virtual machine or container has double vCPU cores. The HEPSPEC06 benchmark is used to calculate starting performance on the physical host. Each configuration is individually evaluated and all VM or container configurations execute with the same benchmark.

The main goal of our test-2 is to compare the improvement in the performance of multiple machines vs. HT on/off. Therefore, all of the input parameters for both experiments were the same. Since the HT enabled machines will give more vCPU cores, these features help us to assign more cores to VMs, which should be evaluated whether or not such features provide meaningful performance improvements.

4.3. Test-3

In the third test case, the design of the previous test cases is applied for physical core isolation and vCPU pinning. To perform the process of isolation, basic Linux commands are used, and the kernel of the host operating system isolates all physical hosts CPU core (excluding one). To get baseline performance, the HEPSPEC06 benchmark is used for executing isolation cores on the cloud's physical hosts. One-to-one strict pinning policy is adopted for pinning the vCPUs of virtual machines or containers. OpenStack supported method for pinning the vCPUs of virtual machine or containers is utilized. As the NUMA node topology can also affect the performance, it is taken under consideration. We executed configuration #2 (which consists of two virtual machines or two containers each having eight vCPU cores). In this configuration, eight vCPU cores of the first virtual machine or container were pinned with NUMA node 0, and second virtual machines or containers

eight vCPU cores were pinned with NUMA node 1. It should be noted that NUMA nodes were assigned to each VM in all of our configurations. At the end, all configurations are run independently on the physical machine of the cloud, which is designed for pinning and isolation purposes. The HEPSPEC06 benchmark runs simultaneously on virtual machines or containers in single configuration.

4.4. Test-4

Similar to test-3, our fourth test case also applied test-2′s configurations, but with the addition of enabling hyperthreading, keeping in mind that the HT-enabled VMs/containers and vCPUs will achieve better performance compared to the previous test cases. Basic Linux commands are used, and the kernel of the host operating system isolates all physical hosts CPU core (excluding one). To get baseline performance, the HEPSPEC06 benchmark is used for executing isolation cores on the cloud's physical hosts. One-to-one strict pinning policy is adopted for pinning the HT-enabled vCPUs of virtual machines and containers.

5. Results and Discussions

Figure 2 comprises the details of all the configurations of the first test. In this result section, performance difference of configurations can be seen clearly. In configuration 1, a single virtual machine or container is assigned with higher number of CPU cores and executes the HEPSPEC06 benchmark. In the result, it can be seen that performance loss of VMs is up to 17% while containers are up to 13%, when only comparing with results of bare metal. Our proposed techniques demonstrate that the overall efficiency of scientific jobs can be maximized when we minimize the number of virtual cores and increase the number of VMs or containers. In Figure 2, configuration 5 can minimize the performance impact as it is the result of 16 virtual machines/container (assigned a single core to each virtual machine/container). As seen, it minimizes the performance lose impact on VMs from 17% to 5% and on containers from 13% to 2%. The other configurations, 2, 3, and 4, can also reduce the overall impact of performance loss. It clearly shows the difference of performance between VMs and containers configuration; containers perform much better than the virtual machine configuration, which is much closer to bare-metal performance. However, it is observed that by using configuration 5, it is possible to achieve near-real-time performance with containers in our cloud environment.

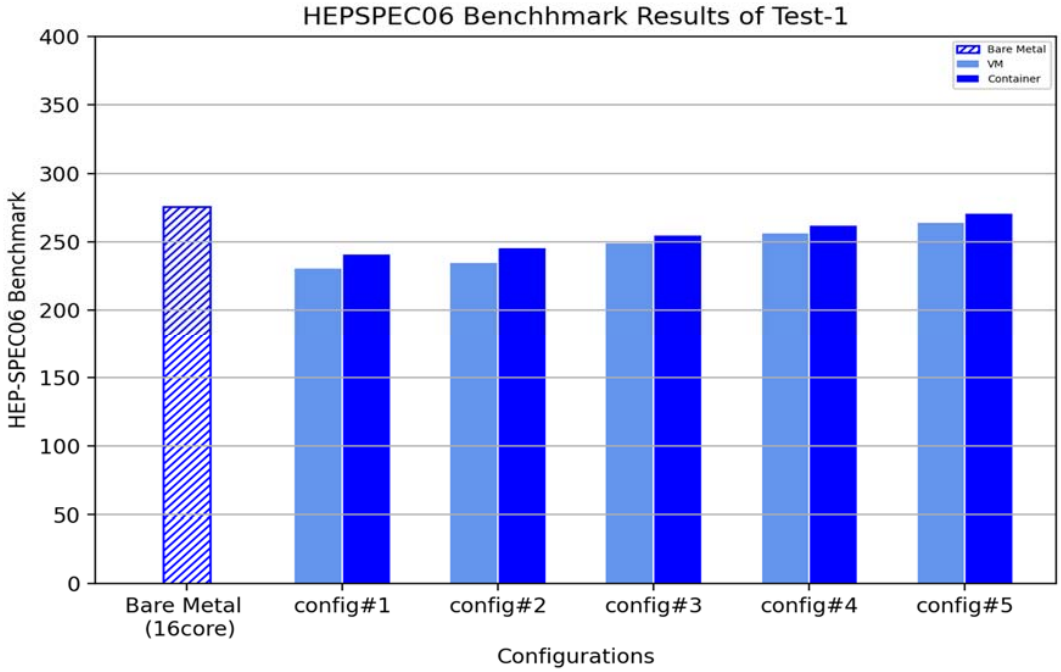

Figure 2. HEPSPEC06 benchmark results for five configurations of test-1.

Figure 3 shows the result of five configurations over test-2. The primary goal of this test was to enhance the overall throughput of the physical system. Since the hyperthreading doubles the number of cores, we often allowed HT on a physical system with 16 cores; it doubles the cores up to 32. As illustrated in Figure 3, the efficiency of HEPSPEC06 on the physical host (bare metal) is improved by up to 16%. This result is compared to the physical host of test-1 having 16 physical cores. For better performance and efficiency, this study used a similar technique of assigning cores per virtual machine or container of the first test case (i.e., test-1) but the total number of cores doubled for each virtual machine or container. As shown in the results, using the HEPSPEC06 benchmark, the performance loss of VM is 18% and containers is up to 14% of configuration 1 compared to the physical machine. With configuration 5, however, we increased the number of virtual machines or containers with decreased numbers of cores, which improved overall performance of VM up to 12% and container up to 10%, while comparing test-2 with configuration 1. The size of the virtual machine or container can vary according to scientific workload so that throughput and performance can be balanced according to requirements.

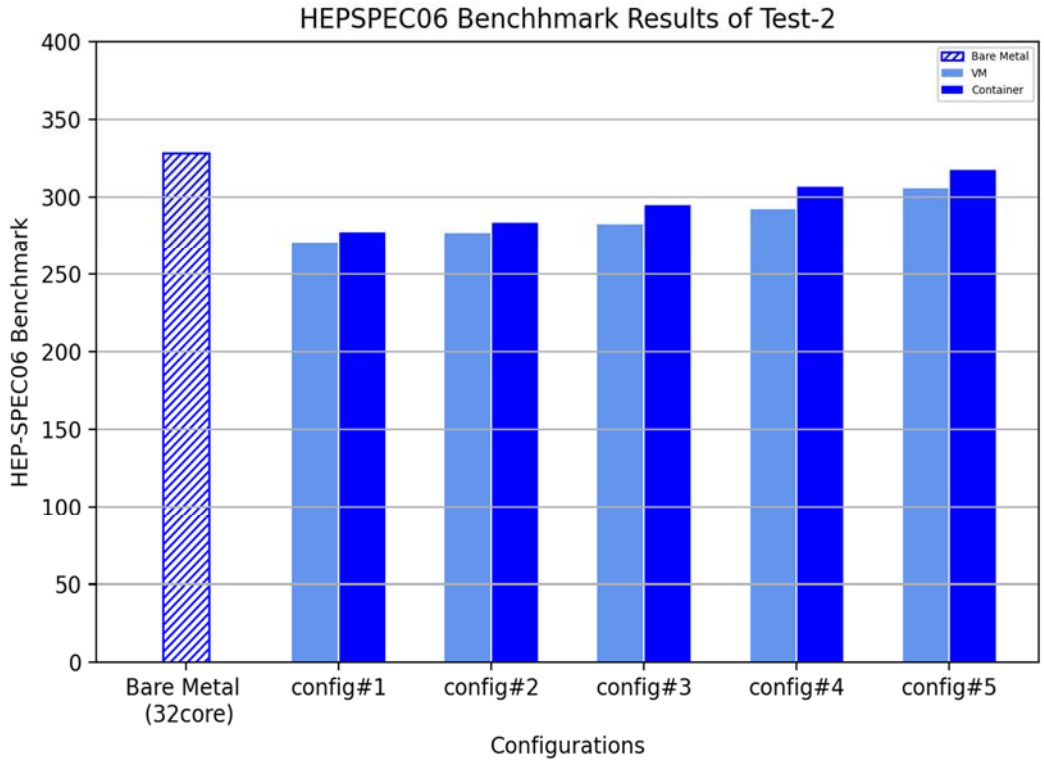

Figure 3. HEPSPEC06 benchmark results for five configurations of test-2.

In test-3, to improve the performance of the VM or container, we applied the pinning and isolation techniques that are implemented for both test cases and consist of five configurations. Figure 4 depicts that the HEPSPEC06 benchmark minimized the overall performance losses; if the number of cores per virtual machine or container decreased and proper pinning and isolation applied, then it further reduces the impact of virtualization from 18% to 6% and of containerization from 14% to 1.5%. It clearly shows that the containers can give much better and near-real-time performance in a cloud environment.

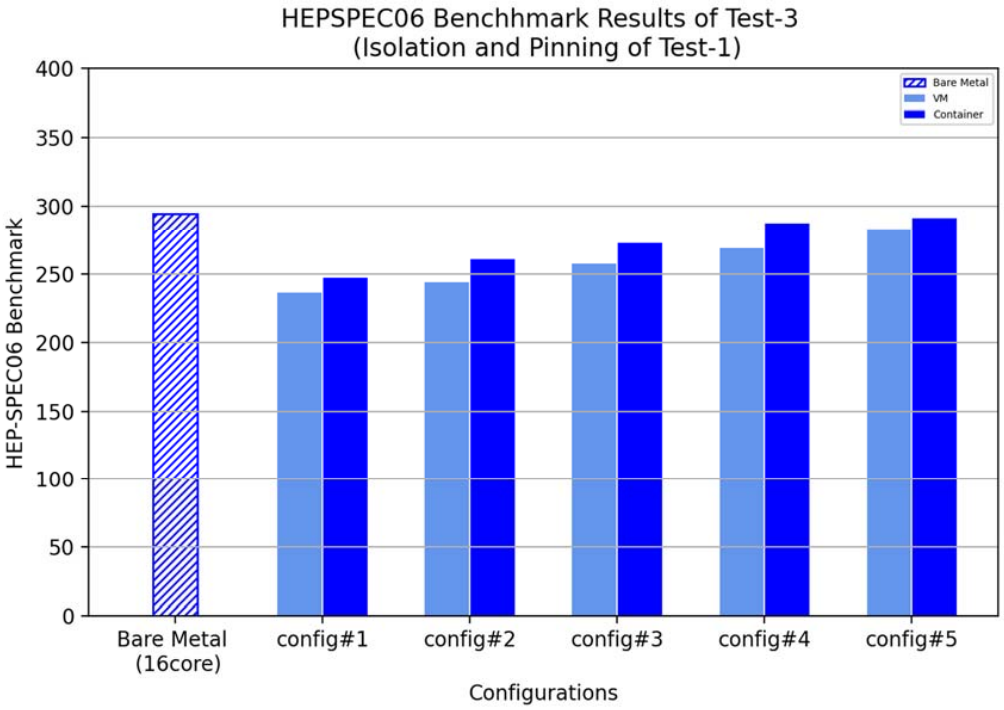

Figure 4. HEPSPEC06 benchmark results of test-3.

The results of the HEPSPEC06 benchmark shows the results of test-4. As illustrated in Figure 5, these results are evaluated after pinning and isolation of HT-enabled VMs/containers and vCPUs for the second test case. The results of test-2 show the identical numbers given by using configuration 5, then configuration 1, and others. We can achieve near-real-time performance of the virtual machine with the difference of only 6% in comparison to bare metal (physical machine). The performance of the container is much closer to bare metal with only a difference of 2%, and it can be considered as near real-time with enabling HT, isolation, and pinning techniques.

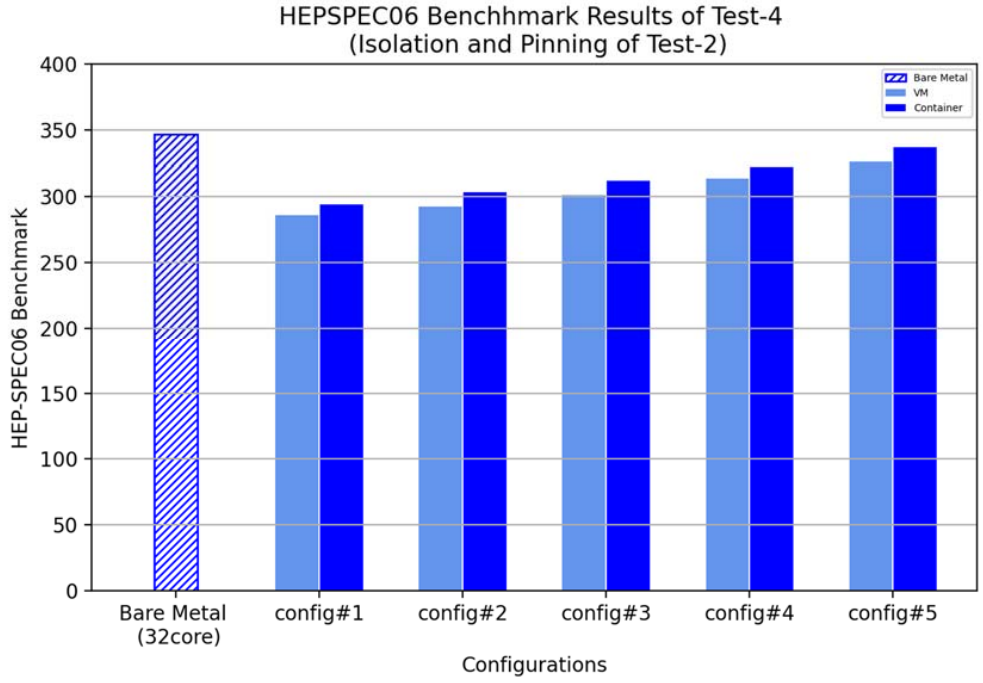

Figure 5. HEPSPEC06 benchmark results of test-4.

Finally, the average performances of virtualization and containerization are tested and comparative analysis of each test case is calculated. As illustrated in Figure 6, test-3, using pinning and isolation with all configurations of second test case (i.e., test-2), can achieve improved efficiency in the cloud environment compared to the other test cases. The average performance of virtual machines increases up to 20% while containers up to 25% compared with the other test cases. This test case can also be used for the scenario with the mixed size of virtual machines or containers required for scientific jobs. In a cloud computing environment, pinning and isolation help to enhance overall throughput. The proper selection mechanism of vCPU cores along with pinning and isolation can improve the throughput and performance of containers as well as VMs in cloud environments for scientific jobs.

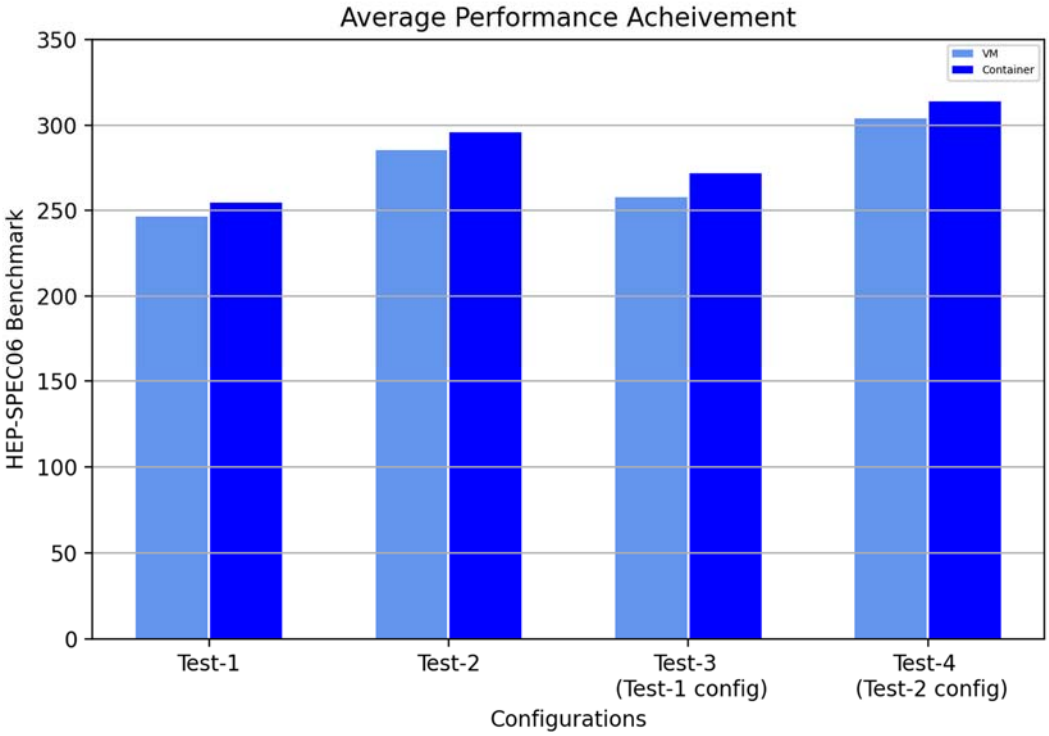

Figure 6. The average VMs performance achievement by each test case.

6. Conclusions

In this paper, we discuss the issues which are related to efficiency and throughput of virtual machines and containers. The performance overhead problems to scientific workloads in a cloud computing environment are evaluated with our configuration scenarios. We propose an approach that combines the four techniques and improves the overall throughput and performance of virtual machines and containers. We present a balanced view of efficiency and throughput for virtualization and containerization. Our analysis and experience can enable VMs or containers to create a cloud-based environment that can deliver the scientific workloads. One of the most promising benchmarks, HEPSPEC06, is used for performance evaluation, enhancements, and accomplishments in cloud computing environments. Our experimental results show that the overall performance and efficiency of VMs and containers can be enhanced only when we choose the minimum cores for a VM or container, and overall throughput of CPU cores is also able to be maximized when we make hyperthreading enabled. Pinning and isolation of physical hosts' CPU cores can further enhance performance in the cloud environment, especially for containerization. This research finds that there is a need for tuning in the virtualization and containerization layers that directly affect the performance in order to achieve the best performance and increase the throughput of virtual machines and containers.

Although our benchmarking is valuable in a data-intensive cloud computing environment, it should be noted that our benchmarking has a limitation. The experimental results described in this paper are evaluating the differences among bare metal, VM, and container in a cloud-based scientific computing environment. Since HEPSPEC06 mainly focuses on science-oriented computing, applications of benchmarking may be limited when comparing with other similar benchmarks that do not seriously consider a data-intensive environment like high energy physics. Therefore, meaningful future research would be to find and evaluate specific working applications that show performance gaps between HEPSPEC06 and other benchmarking tools.

Author Contributions: Conceptualization, S.A.R.S.; methodology, S.A.R.S., T.-H.K., and S.-Y.N.; software, S.A.R.S. and A.W.; validation, S.A.R.S. and H.Y.; formal analysis, S.A.R.S. and A.W.; resources, S.-Y.N. and H.Y.; writing—original draft preparation, S.A.R.S. and A.W.; writing—review and editing, S.-Y.N., M.-H.K., T.-H.K., and H.Y.; supervision, S.-Y.N. and T.-H.K.; funding acquisition, S.-Y.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2008-00458) and Korea Institute of Science and Technology Information.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to extend our sincere thanks to Global Science experimental Data hub Center at Korea Institute of Science Technology Information for their supporting our research and providing the experimental environment.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Armbrust, M.; Fox, A.; Griffith, R.; Anthony, D.J.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; et al. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
2. Ostermann, S.; Iosup, A.; Yigitbasi, N.; Prodan, R.; Fahringer, T.; Epema, D. A performance analysis of EC2 cloud computing services for scientific computing. In *Cloud Computing*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 115–131.
3. Foster, I.; Zhao, Y.; Raicu, I.; Lu, S. Cloud computing and grid computing 360-degree compared. In Proceedings of the 2008 Grid Computing Environments Workshop, Austin, TX, USA, 12–16 November 2008; pp. 1–10.
4. Wang, L.; Zhan, J.; Shi, W.; Liang, Y.; Yuan, L. In cloud, do mtc or htc service providers benefit from the economies of scale? In Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers, Portland, OR, USA, 16 November 2009; Association for Computing Machinery: New York, NY, USA, 2009; p. 7.
5. Beloglazov, A.; Buyya, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurr. Comput. Pract. Exp.* **2012**, *24*, 1397–1420. [CrossRef]
6. Vecchiola, C.; Pandey, S.; Buyya, R. High-performance cloud computing: A view of scientific applications. In Proceedings of the 10th International Symposium on Pervasive Systems, Algorithms, and Networks (ISPAN), Kaohsiung, Taiwan, 14–16 December 2009; pp. 4–16.
7. OpenStack an Open Source Cloud Management Platform. Available online: https://www.openstack.org (accessed on 12 December 2014).
8. HEPSPEC06 Benchmark. Available online: https://w3.hepix.org/benchmarking (accessed on 15 October 2015).
9. Rao, J.; Wang, K.; Zhou, X.; Xu, C.Z. Optimizing virtual machine scheduling in NUMA multicore systems. In Proceedings of the IEEE 19th International Symposium on High Performance Computer Architecture (HPCA2013), Shenzhen, China, 23–27 February 2013; pp. 306–317.
10. Ibrahim, K.Z.; Hofmeyr, S.; Iancu, C. Characterizing the Performance of Parallel Applications on Multi-socket Virtual Machines. In Proceedings of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, USA, 23–26 May 2011; pp. 1–12.
11. Mao, M.; Humphrey, M. A performance study on the vm startup time in the cloud. In Proceedings of the IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, USA, 24–29 June 2012.
12. Xu, F.; Liu, F.; Jin, H.; Vasilakos, A.V. Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. *Proc. IEEE* **2013**, *102*, 11–31. [CrossRef]

13. Podzimek, A.; Bulej, L.; Chen, L.; Binder, W.; Tuma, P. Analyzing the Impact of CPU Pinning and Partial CPU Loads on Performance and Energy Efficiency. In Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), Shenzhen, China, 4–7 May 2015; pp. 1–10.
14. Varrette, S.; Plugaru, V.; Guzek, M.; Besseron, X.; Bouvry, P. HPC Performance and Energy-Efficiency of the OpenStack Cloud Middleware. In Proceedings of the 43rd International Conference on Parallel Processing Workshops (ICCPW), Minneapolis, MN, USA, 9–12 September 2014; pp. 419–428.
15. High-Performance Linpack (HPL). Available online: http://www.netlib.org/benchmark/hpl/ (accessed on 8 July 2020).
16. Graph500. Large Scale Benchmarks. Available online: https://graph500.org/ (accessed on 1 August 2020).
17. Wang, L.; Kunze, M.; Tao, J. Performance evaluation of virtual machine based Grid workflow system. *Concurr. Comput. Pract. Exp.* **2008**, *20*, 1759–1771. [CrossRef]
18. Jackson, K.R.; Ramakrishnan, L.; Muriki, K.; Canon, S.; Cholia, S.; Shalf, J.; Wasserman, H.J.; Wright, N.J. Performance analysis of high performance computing applications on the amazon web services cloud. In Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), Indianapolis, IN, USA, 30 November–3 December 2010; pp. 159–168.
19. Barik, R.K.; Lenka, R.K.; Rao, K.R.; Ghose, D. Performance analysis of virtual machines and containers in cloud computing. In Proceedings of the International Conference on Computing, Communication and Automation (ICCCA), Noida, India, 29–30 April 2016.
20. Yan, J.; Zhang, H.; Xu, H.; Zhang, Z. Discrete pso-based workload optimization in virtual machine placement. *Pers. Ubiquitous Comput.* **2018**, *22*, 589–596. [CrossRef]
21. Mavridis, I.; Karatza, H. Combining containers and virtual machines to enhance isolation and extend functionality on cloud computing. *Future Gener. Comput. Syst.* **2019**, *94*, 674–696. [CrossRef]
22. Chae, M.; Lee, H.; Lee, K. A performance comparison of linux containers and virtual machines using Docker and KVM. *Clust. Comput.* **2019**, *22*, 1765–1775. [CrossRef]
23. Bui, B.; Mvondo, D.; Teabe, B.; Jiokeng, K.; Wapet, L.; Tchana, A.; Depalma, N. When extended para-virtualization (xpv) meets numa. In Proceedings of the Fourteenth EuroSys Conference, Dresden, Germany, 25–28 March 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1–15.
24. Li, J.; Qian, J.; Guan, H. A Holistic Model for Performance Prediction and Optimization on NUMA-based Virtualized Systems. In Proceedings of the INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 352–360.
25. Intel Hyper-Threading Technology: Technical User Guide 2013. Available online: https://www.utdallas.edu/~{}edsha/parallel/2010S/Intel-HyperThreads.pdf (accessed on 5 April 2020).
26. Integer Component of SPEC CPU2000. Available online: https://www.spec.org/cpu2000/CINT2000/ (accessed on 10 March 2020).