International Journal of Computing Science and Information Technology, **2018,**
ISSN: 2278-9669, April 2018 (http://ijcsit.org)

A Methodical Review Of Virtualization Techniques In Cloud Computing

Salini Suresh [1].
Seshadripuram Academy of Business Studies,
Bangalore, India;

Manjunatha Rao [2].
Dr. Ambedkar Institute of Technology,
Bangalore, India,

Received January 2018

Abstract

IT enterprises has widely embraced virtualization trend to incorporate autonomic computing. Server virtualization empowers the creation of virtual environments that are adept to run operating system and applications. Recent IDC survey forecasts that companies can achieve a return of investment of 472 percent by utilizing virtualization. We have presented a detailed overview of Hypervisor based and container based implementations. We reviewed the existing approaches of performance evaluation and analysis of the virtualisation tools.

Keywords: Virtualization, Hypervisor, VM, Container

I. Introduction

Virtualization can be defined as "framework or methodology of dividing the resources of a computer into multiple implementation environments, by concerning one or more notions or technologies such as software partitioning, hardware division, time-sharing, partial or complete machine simulation, emulation, worth of service, and many others"[1]. Server virtualization facilitates the creation of several isolated virtual instances from one physical server. Virtualization conceals the identity of server resources from server users and each virtual server will performance like a physical server. Network virtualization pools the accessible assets of a system to the available bandwidth into independent channels that can be allocated to a particular server. Storage virtualization merge physical storage from many devices to a single storage pool.
Cloud implementation benefits largely by virtualization as it allows the creation of virtual instances of physical resources [2]. The core paybacks of virtualization are hardware independence, multitenancy, flexibility, increased isolation, and enhanced security. The virtualization systems are categorized into are full virtualization, Para virtualization, and operating system level virtualization. In Full virtualization operating systems and applications are isolated from the computer hardware by a software called Hypervisor thus provides each Virtual machine (VM) a complete abstraction VMware [3] and KVM [4] uses this type of hardware virtualization. Xen [3] uses Para virtualization where hypervisor changes the guest in an operating system in contrast to virtual machine model, which allows a guest operating system to run without modifications. Operating System Level virtualization or container-based virtualization creates user instances or containers isolated from each other. Linux-VServer [5], OpenVZ [6] and Linux Containers (LXC) [7]) practices container-based virtualization, which is lightweight compared to VM based model. The rest of the manuscript is organised as follows: Section II describes background and related work.
Section III discusses an overview of virtualisation techniques. In Section IV performance analysis of virtualisation techniques is in existing approaches are reviewed.

II. BACK GROUND AND RELATED WORK

Most of the common server operating systems lack configuration isolation, which requires the installation of a separate instance of the operating system for each application in a virtual machine. Virtualization techniques should provide the resource isolation in a cloud framework [8]. Hypervisor-based systems are the most accepted solution for cloud virtualization [9].Bare metal hypervisor or Type-1 Hypervisors can be mounted on the hardware directly whereas hosted hypervisor or Type-2 hypervisor entails a host operating system. Kernel Virtual Machine (KVM) is a virtualization tool for Linux [10].

A guest operating system thus runs on another level above the hypervisor, allowing for true isolation of each virtual machine. This is the classic VM architecture. An example of this implementation is the VMware ESX Server and Xen. Linux-VServer [11], OpenVZ [12] and Linux Containers (LXC) [13] offer container based implementations .While Hypervisors provide abstraction at the hardware level , containers provide abstraction at the application level. Generally, container-based virtualization systems use Control Groups (Cgroup) [14] to manage resources like CPU, memory. Figures 1(a) and 1(b) illustrates Hypervisor based and container-based virtualization.

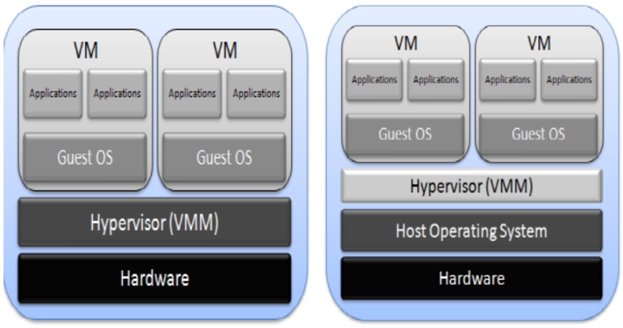

Figure1: Hypervisor based and container-based virtualization.

III. OVERVIEW OF VIRTUALISATION TECHNIQUES.

In this section, an overview of virtualisation tools are given.

A. KVM

KVM (Kernel-based Virtual Machine) is a feature of Linux that uses full virtualization by], installing an unmodified guest operating system (OS) for Linux process. KVM provisions both emulated I/O devices through QEMU [15] and paravirtual I/O devices using virtio [16]. The full virtualized system provide excellent isolation but overhead associated with data transfer is high as creation and deployment of disk images for access to storage is time-consuming, leading to wastage of storage space. In antagonistic to the prevalent conviction that Hypervisor -based systems have high-performance overhead, studies show that performance overhead of VM could vary on the job-to-job basis [17].

B. LXC

Containerization technology lays its base on LXC (Linux containers).Lightweight isolation is usually implemented by using Kernel Cgroups and namespaces. Namespaces provide resource isolation, network isolation while Cgroups manage the resources, process control and define the configuration of a network. LXC has also taken on of Linux Kernel features like chroot, PID and enable users to create and manage applications through APIs[18].

C. DOCKER

Docker is lightweight virtualization technology based on LXC containers, which encapsulate application and all its dependencies in a container, hence with less overhead than a VM. This Linux based open platform facilitates developers for rapid development and execution of applications, along with the set of tool and workflows for easier deployment and management[19].

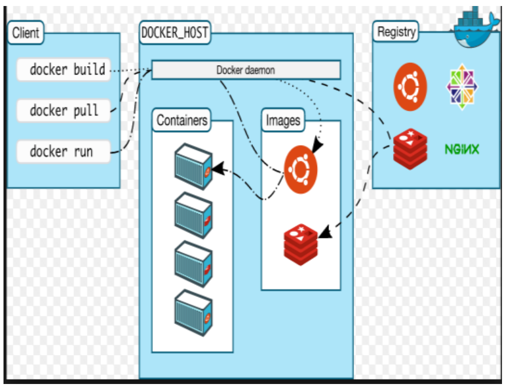

Figure 2 demonstrates an architectural overview of Docker.

A Docker image basic build component or read-only template to which other components like middleware, web application etc. can be added. Docker registries are repositories to share the images [20].

IV. PERFORMANCE ANALYSIS OF VIRTUALISATION TOOLS.

In this section, we reviewed the existing approaches for performance evaluation and analysis of the virtualisation tools.

Sudha et al. [21] have compared the performance of Hypervisor – based system (KVM) and container-based system (LXC) and have conducted computation and I/O performance test with Apache Benchmark and IO zone File system. They have analysed that LXC containers out-performed KVM and showed a performance nearer to that of the host operating system.

David Beserra et al. [22] studies the effect of resource sharing among containers in performance of I/O-bound applications in virtualized environments. They observe that physical resources when shared among are shared between logically different platforms, LXC and Docker performs as good as non-virtualized environments but both the tools do not provide adequate isolation for the resources.

Miguel G. Xavier et al. [23] investigated that even though container based system offers near-native performance, they lack isolation and security.

Jyothi Shetty et al.[24] evaluates overhead of virtualization with benchmark tests that contain CPU, system, memory and disk workloads using Phoronix test suite for Docker containers in comparison with VM, and finds Docker outdo VM in terms of CPU and memory workloads, the disk I/O performance. Conversely, the hitch of containers is the isolation and security level because containers share the host kernel.

Li et al. [25] evaluated the I/O bandwidth and latency reduction in virtual machine hosts and storage servers using real world trace driven simulation. They have proposed a procedure, Sea-Cache, which fuses de-duplication and a storage server side de-duplication system that in turn optimize storage server to host data transfers and host side caching.

Sungyong Ahn et al. [26] have examined the performance of Solid-State Drives (SSDs) in the virtualized Hadoop through various benchmark tests and observed that the increased I/O overhead due to virtualization is mainly responsible for the performance degradation, while the CPU overhead is negligible and virtualization generates fragmented and randomized I/O workload pattern producing additional I/O overhead. I/O workload bothered by virtualization.

Charles F. Go calves et al. proposed a security benchmark to assess and compare various virtualized systems to cloud providers to choose the right set-up. The evaluation procedure investigates presence of vulnerabilities and, assessing trustworthiness of the system.

V. CONCLUSION

Virtualization has greatly transformed the information technology (IT) industry in diverse areas such as network, operating systems, applications or storage. Our study on the existing approaches of performance evaluation and analysis of the virtualisation tools shows that both hypervisor based and container based virtualisation had increased the efficiency than non-virtualized environments.

REFERENCES:

[1] A Survey on Security Aspects of Server Virtualization in Cloud Computing O Sri Nagesh, Tapas Kumar, Vedula Venkateswararao, International Journal of Electrical and Computer Engineering (IJECE) Vol. 7, No. 3, June 2017, pp. 1326~1336.

[2] 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.

[3] “Xen,” 2012. [Online]. Available: http://www.xen.org.

[4] “KVM,” 2012. [Online]. Available: http://www.linux-kvm.org

[5] “Linux VServer,” 2012. [Online]. Available: http://linux-vserver.org

[6] “OpenVZ,” 2012. [Online]. Available: http://www.openvz.org

[7] “Linux Containers,” 2012. [Online]. Available: http://lxc.sourceforge.net.

[8] An updated performance comparison of virtual machines a nd Linux containers Wes Felter; Alexandre Ferreira; Ram Rajamony; Juan Rubio,2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2015,pp. 171 - 172

[9] X. Xu, H. Yu, and X. Pei, “A novel resource scheduling approach incontainer based clouds,” in *Proc. 17th IEEE Int. Conf. Comput. Sci.Eng. (CSE 2014)*. Chengdu, China: IEEE Computer Society, Dec. 2014, pp. 257–264.

[10] “Linux Containers,” 2012. [Online]. Available: http://lxc.sourceforge.net

[11] Avi Kivity, Yaniv Kamay, Dor Laor, Uri Lublin, and Anthony Liguori.KVM: the Linux virtual machine monitor. In Proceedings of the Linux Symposium, volume 1, pages 225–230, Ottawa, Ontario, Canada, June 2007.

[12] “Linux VServer,” 2012. [Online]. Available: http://linux-vserver.org

[13] “OpenVZ,” 2012. [Online]. Available: http://www.openvz.org.

[14] “Linux Containers,” 2012. [Online]. Available: http://lxc.sourceforge.net.

[15] “Control groups definition, implementation details, examples and api,” 2012. [Online]. Available: http://www.kernel.org/ doc/Documentation/ cgroups /cgroups.txt.

[16] Fabrice Bell rd. QEMU, a fast and portable dynamic translator. In Proceedings of the Annual Conference on USENIX Annual Technical Conference, ATEC ’05, pages 41–41, Berkeley, CA, USA, 2005.USENIX Association.

[17] Rusty Russell. Virtio: Towards a de-facto standard for virtual I/O devices. SIGOPS Oper. Syst. Rev., 42(5):95–103, July 2008.

[18] S. Soltesz, H. P¨tzl, 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.

[19] Performance Overhead Comparison between Hypervisor and Container Based Virtualization Zheng Li; Maria Kihl; Qinghua Lu; Jens A. Andersson 2017, IEEE 31st International Conference on Advanced Information Networking and Applications (AINA),Year: 2017.

[20] “About Docker” Docker documentation. https://docs.docker.com/engine/misc/.

[21] *Performance Analysis of Linux Containers - An Alternative Approach to Virtual Machines Sudha et al., International Journal of Advanced Research in Computer Science and Software Engineering 4(1), January - 2014, pp. 820-824 © 2014, IJARCSSE*

[22] Comparing the Performance of OS-level Virtualization Tools in SoC-based Systems: the caseof I/O-bound Applications David Beserra_y, Manuele Kirsch Pinheiroy 2017, IEEE Symposium on Computers and Communications (ISCC).

[23] Performance evaluation of container based virtualization for High Perfor-

mance Computing Environments Miguel G. Xavier; Marcelo V. Neves; Fabio D. Rossi; Tiago C. Ferreto; Timoteo Lange; Cesar A. F. De Rose2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing ,IEEE, Year: 2013,Pages: 233 - 240

[24] An Empirical Performance Evaluation of Docker Container, Openstack Virtual Machine and Bare Metal Server Jyoti Shetty*1, Sahana Upadhaya2, Rajarajeshwari HS3, Shobha G4, Jayant Chandra5, Indonesian Journal of Electrical Engineering and Computer Science Vol. 7, No. 1, July 2017, pp. 205 ~ 213

[25] M. Li, S. Gaonkar, A. R. Butt, D. K. K. Voruganti, *"Co-operative Storage-Level De-Duplication for I/O Reduction in Virtualized Data Centers"* in IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and telecommunication Systems, 2012

[26] S. Ahn, S. Park, J. Hong and W. Chang, *"Performance Implications of SSDs in Virtualized Hadoop Clusters"* in IEEE International Congress on Big Data, 2014 .

Benchmarking the Security of Virtualization Infrastructures: Motivation and ApproachCharles Ferreira Gonçalves2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)Year: 2017Pages: 100 – 103.