PERFORMANCE ANALYSIS OF KERNEL-BASED VIRTUAL MACHINE

Sudha M[1], Harish G M[2], Nandan A[3], Usha J[4]

[1]Department of MCA, R V College of Engineering, Bangalore : 560059, India
sudha.mooki@gmail.com
[2]Department of MCA, R V College of Engineering, Bangalore : 560059, India
harish.gm@gmail.com
[3]Department of MCA, R V College of Engineering, Bangalore : 560059, India
nandankengeri@gmail.com
[4]Department of MCA, R V College of Engineering, Bangalore : 560059, India
ushajayadevappa@yahoo.com

ABSTRACT

Rapid advancement in computing technology has put forth Cloud computing as a paramount paradigm in distributed systems. It is very much essential and important too to fully understand the underlying technologies that makes clouds possible. One key technology that make makes the cloud popular is virtualization. Even though virtualization technology is not new, the concept of hypervisors with virtualization is getting popular and also well understood by many. There are a good number of hypervisors. This paper discusses the types of Virtualization Technologies, hypervisors and configuration of the VMs and analyse the performance of Virtual Machines using Kernel Based Virtual Machine- KVM, a Type2 hypervisor.

KEYWORDS

Virtual Machines, Hypervisor, Virtualization, Processor Pinning, KVM

1. INTRODUCTION

With technological advancements CPUs with Multi-cores seemly are the de facto standard. Every single core in a multi-core CPU has the power to run a physical machine. This can be achieved through virtualization. The concept of virtualization is not really new. The recent demands, such as the need for increased hardware utilization, reduced hardware costs, lower power consumption, on-demand access have led to a drastic deployment of virtualization techniques and solutions. The conceptual idea behind virtualization was to allow multiple and different parallel instances of the operating system on a physical machine. Every instance share the resources available at the host machine. The resources can be the CPU, memory, cache, network etc. [1]. Hence a virtual machine (VM) is "completely separated guest operating system installed within or upon a host operating system"[2]

2. VIRTUAL MACHINES

Popek and Goldberg [3 have defined VM as "an efficient, isolated duplicate of a real machine". Presently virtual machines which have no direct interaction any of the real hardware is popular. A virtual machine (VM) is a software that emulates the physical machine. Virtual machines are categorized into two groups as Process Virtual Machine and System Virtual Machine.

2.1 Process Virtual Machines

A process virtual machine or application Virtual machine is designed to run a single program with a single process. It runs just like a regular application within the host OS as a process. The VM is created when process is initiated and destroyed when the process exits or dies. A Process VM is sometimes referred to as application virtual machine. This VM mainly aims at providing a platform-independent development environment. Java programming language is platform independent as it implements Java Virtual Machine (JVM) which is a process VM.

2.2 System Virtual Machines

A System Virtual Machine gives a complete virtual hardware platform with support for execution of a complete operating system (OS).

The advantage of using System VM are [4] :

- **M**ultiple Operating System environments can run in parallel on the same piece of hardware in strong isolation from each other.
- The VM can provide an instruction set architecture (ISA) that is slightly different from that of the real machine

The main draw backs are:

- Since the VM indirectly accesses the same hardware the efficiency is compromised.
- Multiply VMs running in parallel on the same physical machine may result in varied performance depending on the workload imposed on the system. Implementing proper isolation techniques may address this drawback.

3. CLASSIFICATION OF VIRTUALIZATION TECHNIQUES

3.1 Hardware or Platform Virtualization

Hardware or platform virtualization aims at creating a VM that behaves or performs just like a real machine with its own operating system. Applications that run on these VM are independent of the resources of the underlying hardware. For example, a computer running Microsoft Windows Operating System may host a virtual machine that runs an Linux Operating system like Ubuntu. [5][6]

Hardware Virtualization is the physical machine is the host machine on which the Virtualization is performed and the Virtual Machine created upon the physical machine or the host, is called the guest machine. The term host and guest are used to distinguish the software that each one of them are running. The software that creates Virtual Machines on the host is known as Hypervisor.

Different types of hardware virtualization include:

3.1.1 Full virtualization

In Full Virtualization the virtual machine is the exact simulation of the physical machine which allows the guest operating system to run unmodified.

3.1.2 Partial virtualization

Partial Virtualization simulates few of the resources of the actual hardware. Hence, some programs on the guest needs to be modified for execution in this environment.

3.1.3 Para virtualization

In this technique the guest programs run within in their own separate region, as though they are isolated. Guest programs needs to be specifically modified to run in this environment.

3.2 Network virtualization

Network Virtualization provides an virtual network environment. Ex. VLANs. A VLAN provides isolation to a group of systems that communicate in the same broadcast domain irrespective of the location of each node. By creating VLANs on the physical hardware, the group of hosts are isolated from others and appears to the remotely located hosts as though they exist on the same physical network. The abstraction provided by VLANs had made it feasible for companies to partition physical connections to define and create less expensive networks and flexible networks to meet the ongoing business demands. [7]

3.3 Application Virtualization:

Application virtualization provides an abstraction in which the traditional applications are wrapped inside a container, thus giving an illusion to the application that it is running on an intended platform. The application considers that it has access to the resources that it needs for execution of the task.

3.4 Desktop virtualization

In desktop virtualization, the traditional desktop is virtualized and moves the work load of the client to the data center. Different means, like thin-client etc., are used to access these workloads.

4. CLASSIFICATION OF HYPERVISORS

Robert P. Goldberg classifies hypervisors into two types [8,9] Type 1 (or native, bare metal) hypervisors run directly on the host's hardware to control the hardware and to manage guest operating systems.

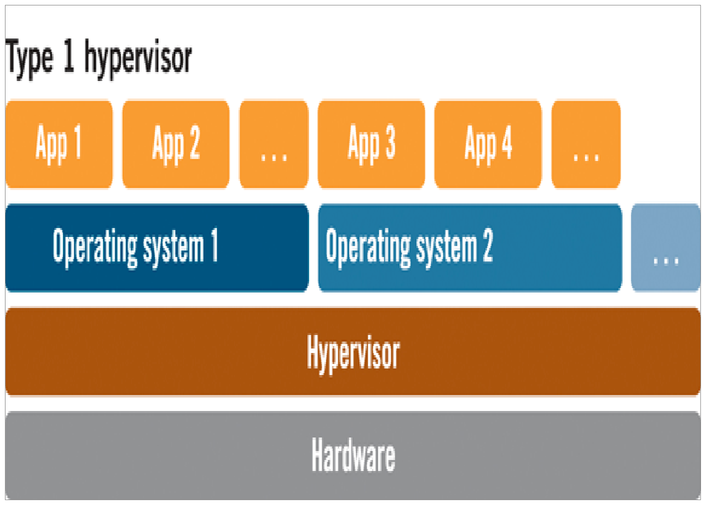

Figure 1. Type1 Hypervisor

A guest operating system thus runs on another level above the hypervisor. To list a few Microsoft Hyper-V, Citrix XenServer, Oracle-VM Server and VMware ESX/ESXi

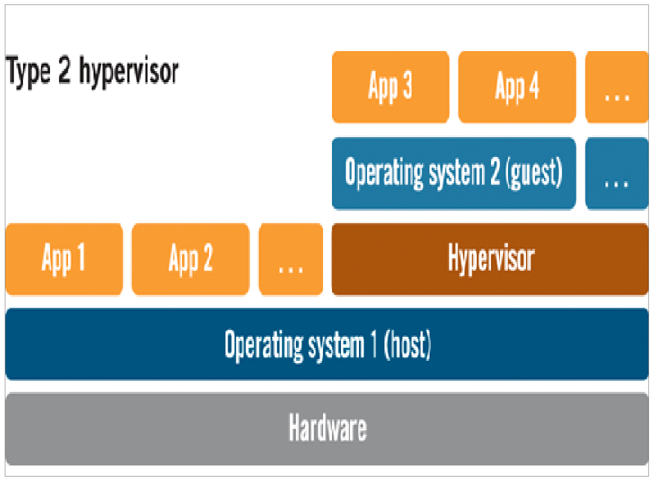

Figure 2. Type2 Hypervisor

Type-2 (or hosted) hypervisors run on top of the conventional operating system environment. The hypervisor layer runs at the level two above the physical hardware and the guest operating systems run at the level three above the physical hardware. Ex. KVM, BHyVe, and VirtualBox. Figure 1 depicts the architecture of Type-1 Hypervisor and Figure 2. depicts the architecture of Type-2 hypervisor. Type-1 hypervisor embedded into the hardware and a Type-2 hypervisor resides above the operating system like Linux.

5. PERFORMANCE TUNING

The virtual machines with type 2 hypervisors can be fine tuned for performance optimization in many ways. One such method is processor pinning.

5.1 Processor Pinning:

In Processor pinning the virtual processors of a guest operating system are pinned to one or more physical processors on the host. By default the processors are not pinned, thus the system can assign any physical processors to any virtual processor in the system. Processor pinning ensures that the system specifically assigns one or more only physical processor to a virtual processor [11]. An application of the guest operating system executes multiple threads for the same task. Since the virtual processors are pinned to the physical processors that share the same cache, the application often looks into the cache for data and instructions for each of the threads.

5.1.1 Drawbacks of Processor pinning :

The major drawback lies in the performance of the VMs due to sharing of the same cache. By pinning the virtual processors executing different tasks share the same underlying physical resource. Ex. A guest operating system. While running any one of the virtual processor the cache will be accessed and will be overwritten. When the other virtual processor runs, and it accesses the same cache and finds no data in the cache**.** The performance of the VM decrease when a virtual processor is pinned to a set of physical processors. One such scenario is the idling of physical processors because they are pinned to workloads that are inactive.

6. PROBLEM STATEMENT

A virtual machine is a replica of the physical computer, as an emulated software. Virtualization provides better utilization of the physical hardware by separating the underlying hardware and the operating system. When multiple VMs are running in parallel on the same host, the performance of each VM may vary and behave differently, because the resources are shared among other VMs. The aim was to compare the Processor performance of type 2 hypervisor under two different configurations and to arrive at an optimal configuration.

6.1 Methodology

The experimental test-bed was set up with the following system configuration:

Hardware

Virtual Machine Configuration : QEMU Virtual 1.0 @ 2.20GHz (1 Core); Motherboard: Bochs; Chipset: Red Hat Virtio; Memory: 1024MB, Disk: 7GB; Graphics: Cirrus Logic GD 5446; Audio: Generic 1af4 ID 20; Network: Red Hat Virtio device4

Software

OS: Ubuntu 12.04 ;Kernel: 3.2.0-23-generic-pae (i686); Desktop: GNOME 3.2.1; Display Server: X Server 1.11.3; Display Driver: cirrus 1.3.2; Compiler: GCC 4.6; File-System: ext4; Screen Resolution 1024x768; System Layer: QEMU 1.0

KVM hypervisor was installed over Ubuntu 12.04 , Kernel-3.2.0-23-generic-pae (i686). Each VM was assigned one core each with 3 VMs running on the Intel Core i3 processor.

The aim was to compare the performance of Type2 Hypervisor – QEMU/KVM which shares the underlying resource, under two configurations (i) without Pinning VM to the core and (ii) By Pinning VM to the core.

We configured the VMs and used Phoronix Test Suite which comes bundled with test cases. We used "Timed Apache Compilation" test case bundled with Phoronix test suite. This test will determine the time taken to build the Apache HTTP Server and analyse the performance of the VMs.

Phoronix Test Suite is a standard open source benchmarking tool for analysing Virtual Machines performance [10]. This is an open-source software licensed under the GNU GPLv3.

It is the most comprehensive benchmarking and testing tool available with an extensible framework with flexibility any ease in adding test cases. This software will effectively carry out quantitative and qualitative benchmarks.

7 RESULTS

The test was performed by compiling the Apache source file in the Virtual Machines with both the configurations. It was found that the performance of the Virtual Machines were better when pinned to the core. The standard deviation and Standard Error results are plotted on a graph and depicted in the figure 3. The standard deviation is 0.8 when the core is pinned and 0.11 when the processor is not pinned. And standard deviation is 0.5 when the core is pinned and 0.6 when core is not pinned. Figure 4 depicts the time taken to compile the Apache source code with both the configurations. The average time taken to compile the Apache HTTP Server source code is 312.42 ms when the core is pinned and 312.96 ms when core is not pinned. The performance of the VM when the core is pinned much more optimal when compared to the other configuration.

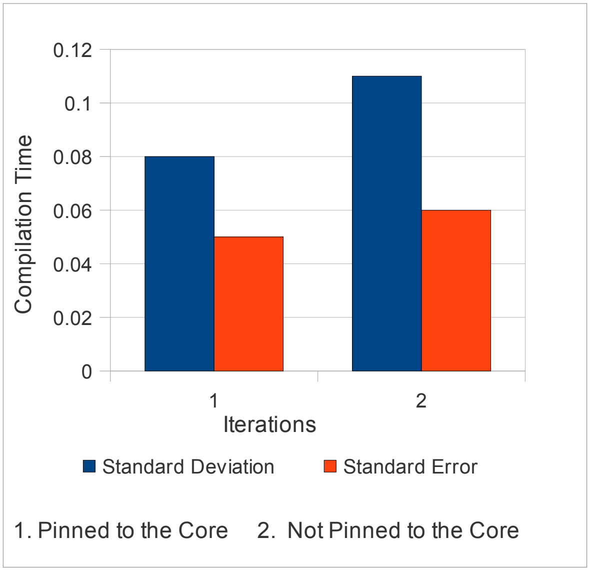

Figure 3 Comparison of different VM Configurations

8. CONCLUSION

Cloud computing has become popular with Virtualization technology. Hypervisors too along with virtualization is getting popular. In this paper we discussed about types of virtualization, hypervisors. Compared the performance of a KVM, a Type2 hypervisor under two configurations, one by pinning VM to the core and the other was by not pinning VM to the core. Phoronix Test Suite's Apache source code compilation test was conducted on both the configurations and it was found that the Virtual Machines performed more efficiently when pinned to the core. With this test results we arrive at a conclusion that applications that demand more CPU time can be pinned to the core for better performance.

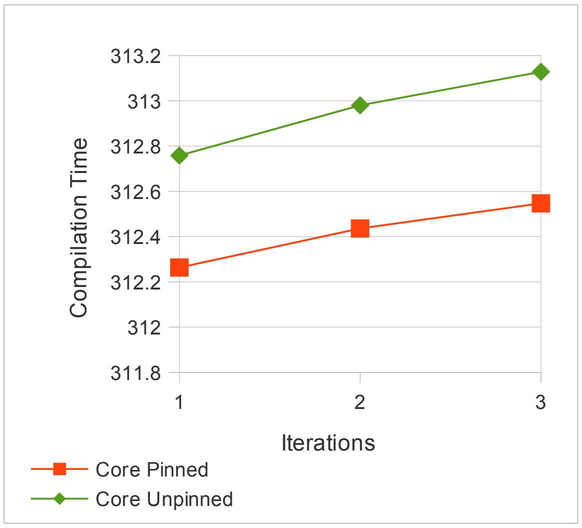

Figure 4 Comparison of Compilation time/iteration

REFERENCES

[1] http://www.virtuatopia.com/index.php/An_Overview_of_Virtualization_and_VMwareServer_ 2.0]

[2] "Virtual Machines: Virtualization vs. Emulation" (http:// www. griffincaprio. com/ blog/ 2006/ 08/virtual-machines-virtualization-vs-emulation.html). . Retrieved 2011-03-11.].

[3] Smith, James; Nair, Ravi (2005). "The Architecture of Virtual Machines". Computer (IEEE Computer Society) 38 (5): 32–38. doi:10.1109/MC.2005.173.

[4] http:/ / www. vmware. com/ solutions/ business- critical-app

[5] Turban, E; King, D; Lee, J; Viehland, D (2008). "Chapter 19: Building E-Commerce Applications and Infrastructure". Electronic Commerce A Managerial Perspective. pp. 27.

[6] "Virtualization in education" (http:/ / www-07. ibm. com/ solutions/ in/ education/ download/Virtualization in Education. pdf). IBM. October 2007.

[7] General Virtualization Articles by Scott.D.Lowe, article published on Sep 08, 2011.

[8] http://en.wikipedia.org/wiki/Hypervisor

[9] Goldberg, Robert P. (February 1973) (PDF). Architectural Principles for Virtual Computer Systems. Harvard University. pp. 22–26. Retrieved 2010-04-12.

[10] www.phoronix-test-suite.com

[11] http://publib.boulder.ibm.com/infocenter/lnxinfo/v3r0m0/index.jsp?topic=%2Fliaat %2Fliaattunpinprocs.htm

AUTHORS

Sudha M, is currently working as Assistant Professor in the department of Master of Computer Applications, R V College of Engineering, Bangalore. She is pursuing her PhD at Visvesvaraya Technological University, Belgaum, Karnataka, India. Her area or research is fault detection in cloud services.

Harish G M, is currently working as Assistant Professor in the department of Master of Computer Applications, R V College of Engineering, Bangalore. His area or interest are Cloud Computing, Information Retrieval and Virtualization.

Nandan A, is currently working as Technical Faculty in the department of Master of Computer Applications, R V College of Engineering, Bangalore. His area or interest includes cluster computing, Linux internals, and virtualization.

Dr. J. Usha is working as Professor in Department of Master of Computer Applications, R.V. College of Engineering, Bangalore, India. She obtained her Ph.D in 2011 from University of Pune, Pune. Her research interests include mobile computing, computer networks and cloud computing. She has over Fourteen years of teaching / research experience. She has published many papers in National / International conferences and journals.