Hypervisor: A Survey on Concepts and Taxonomy

Ankita Desai, Rachana Oza, Pratik Sharma, Bhautik Patel

Abstract*— Because of the advancement of VLSI technology and aggregate throughput of all devices of servers, we are having tremendous computing power which could not be utilized either 100% or optimized way. Virtualization technique has solved this problem by providing proper utilization of hardware resources. Virtualization refers to the abstraction of computer resources. It separates user and applications from the specific hardware characteristics they use to perform their task and thus creates virtual environment. The purpose of creating virtual environment is to improve resource utilization by aggregating heterogeneous and autonomous resources. This can be provided by adding a layer called HYPERVISOR between OS and underlying hardware. There are many market players who have launched the hypervisor. This paper mentions architectural specification of each hypervisor followed by common characteristics that each hypervisor poses.

***Index Terms*— AMD, Cloud Computing, Hypervisor, Intel VT-x, Virtualization.**

I. INTRODUCTION

Cloud computing is transforming the way people use computers as well as how networked services are run. The cloud service provider is able to dynamically provide infrastructure to meet the current demand by leasing infrastructure from cloud infrastructure provider. The provider does this by utilizing virtualization where virtual machines from multiple customers share the same physical server.

Virtualization has transformed the thinking from physical to logical, treating IT resources as logical resources rather than separate physical resources. To provide virtualization, *hypervisor* become the fundamental software, which is also known as Virtual Machine Monitor (VMM). Its primary responsibility is to monitor the Virtual Machines (VMs) that are running on top of it. There are many players in cloud computing market who are providing this technology like Red Hat, VMware, xen.org, Microsoft etc. This paper will show how these market players have defined hypervisor [3].

The hypervisor is the software layer that abstracts the hardware from the operating system permitting multiple operating systems to run on the same hardware. The hypervisor runs on the host system allowing virtual machines to run on the host's hardware as well. - ***Red Hat***

Revised Manuscript Received on February 06, 2013.

Ankita Desai, Computer Engineering Department, CHARUSAT, Changa, India

Rachana Oza, Computer Engineering Department, CHARUSAT, Changa, India.

Pratik Sharma, Computer Engineering Department, CHARUSAT, Changa, India.

Bhautik Patel, Computer Engineering Department, CHARUSAT, Changa, India.

A thin layer of software, that generally provides virtual partitioning capabilities which directly run on hardware, but underneath higher-level virtualization services. Sometimes it is referred to as a "bare metal" approach. – ***VMware***

An alternative term for virtual machine manager, used because it means 'beyond supervisor', since it is responsible for managing multiple 'supervisor' kernels. - ***xen.org***

From the above definitions, this paper has defined *Hypervisor* in following manner:

"Hypervisor is a thin software layer that provides abstraction of hardware to the operating system by allowing multiple operating system or multiple instances of the same operating system, termed as *guests*, to run on a *host* computer."

II. BACKGROUND

To take the benefit of Virtualization, hardware assistant is required. For that Intel and AMD has created new processor extension to x86 architecture, which is known as Intel VT-x and AMD-VT respectively. Virtualization takes benefit of x86 architecture for providing protected mode virtualization. The Intel 80286 chipset has introduced two methods of addressing memory: real mode and protected mode. The protected mode provides many features in order to support multicasting, such as hardware support for virtual memory and segmenting processor [2].VMM can be classified in two types as shown in Fig. 1:

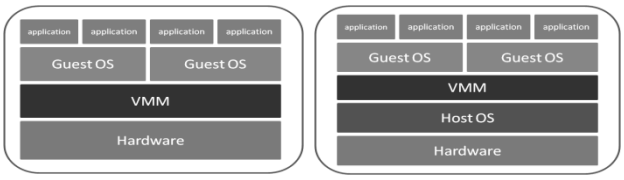
A. Type I Hypervisor

It is also known as "native hypervisor" or "bare metal" as it directly runs on the top of the underlying hardware as shown in Fig. 2. In this case VMM is a small code whose responsibility is to schedule and allocate system resources to VMs as there is no operating system running below it [12]. Examples of Type I VMM are VMware ESX and Xen. In this case the VMM provides device drivers that guest OS uses to directly access the underlying hardware [4].

B. Type II Hypervisor

It is also known as hosted VMM as hypervisor runs as an application in a normal operating system that is known as host operating system. The host OS does not have any knowledge about Type II VMM; it treats it as any other process. It typically performs I/O on behalf of guest OS. The guest OS issues the I/O request that is trap by host OS that in turn send to device driver that perform I/O. The completed I/O request is again route back to guest OS via host OS [4].

As seen above, there are many organization are providing virtualization through hypervisor. In the following section we will see how they are doing this follow by its comparative analysis.



a. Type I Hypervisor b. Type II Hypervisor

Fig1: Types of Hypervisor

III. COMMERCIAL PRODUCTS

In this section the survey of dominant hypervisor is presented:

A. Hyper V

Windows hyper-v is a hypervisor based virtualization system for x86-64 architecture. That provides reliable virtualization environment and integrated management that enable customer to virtualized their infrastructure and reduce cost. The architecture of the Hyper-V is shown in the Fig. 2:

In Microsoft hyper-V, the primary responsibility of hypervisor is to provide isolated execution environment that is known as partition [6]. The partition is a logical unit, in which operating system execute. The hypervisor must have one root or parent partition running windows server 2008 which then create child partition via hypercall application programming interface that host the guest operating system.

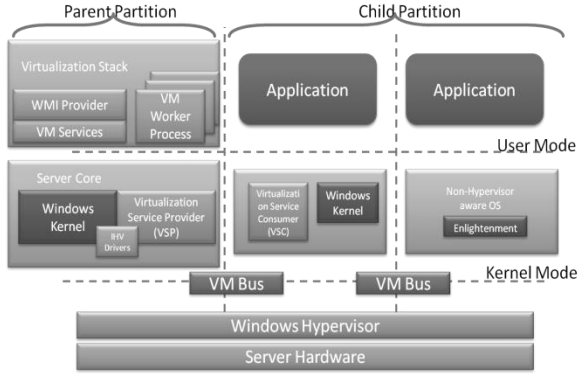


Fig 2: Hyper-V Architecture

The partitions do not have direct access to the physical processor, even they do not handle processor interrupt all they have is the virtualized view of the processor that runs in the virtual memory address space that is private to each guest partition. The processor interrupt is handling by hypervisor which in turn redirect them to respective partition. Child partitions also do not have direct access to other hardware resources. They are presented a virtual view of those resources, as virtual devices (VDevs). Requests to the virtual devices are redirected either via the VMBus or the hypervisor to the devices in the parent partition, which handles the requests. The VMBus is a logical inter-partition communication channel. The parent partition hosts Virtualization Service Providers (VSPs) which communicate over the VMBus to handle device access requests from child partitions. Child partitions host Virtualization Service Consumers (VSCs) which redirect device requests to VSPs in the parent partition via the VMBus. This entire process is transparent to the guest operating system.

The parent partition is the only partition that has direct access to physical memory or devices. To create and configure virtual machines and to provide I/O management and to do other such task parent partition has virtualization stack- a collection of software components that together work to provide such support to virtual machine.

The virtualization stack contains WMI Provider, VM Services and VM Worker Process. VM Worker Process is the user mode component of virtualization stack [8]. The worker process provides virtual machine management services from the Windows Server 2008 instance in the parent partition to the guest operating systems in the child partitions. The Virtual Machine Management Service spawns a separate worker process for each running virtual machine. The Virtual Machine Management Service exposes a set of Windows Management Instrumentation (WMI)-based APIs for managing and controlling virtual machines.

B. KVM

KVM stands for Kernel-based Virtual Machine that is implemented as a loadable kernel module that converts the Linux kernel into bare-metal hypervisor [7]. KVM does not run on CPUs without the hardware virtualization extensions like Intel VT-X and AMD. KVM consist of two modules:

Kvm.ko: A loadable lernel module that provides the core virtualization infrastructure.

Kvm-[intel|amd].ko: A processor specific module for Intel or AMD.

In KVM architecture, each virtual machine is implemented and treated as regular Linux process. This allows KVM to take benefits of all Linux Kernel feature. Actual these feature-VMs are treated as a normal Linux Process and can take benefits of Linux kernel-has made KVM mature rapidly and over-take other open-source hypervisor.

In addition, it uses QEMU to emulate motherboard hardware like, memory controller, network interface, ROM BIOS etc [2]. QEMU is a machine emulator that runs an unmodified target operating system and all its application in virtual machine. The primary usage of QEMU is to run one operating system on another i.e. Windows on Linux or Linux on Windows [1].

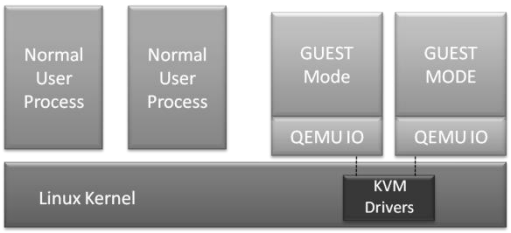


Fig 3: KVM Architecture

As stated earlier, KVM inherits memory management features of Linux Kernel. The memory for virtual machine is stored as memory for any other Linux process and can be swapped, backed by large pages for better performance, shared or backed by a disk file. Memory page sharing is supported by a kernel feature called Kernel Same Page Merging (KSM). It scans the memory of each virtual machine and wherever it finds two VMs have same page it merge those pages, store it as single copy and made is shared between

VMs. Whenever a guest OS wants to modify that page it is provided its own copy. KVM is able to any storage and any hardware system supported by Linux Kernel. Any upgradation to any of this technique can be easily accommodated by KVM.

C. XEN

Xen hypervisor is software layer that directly sits on the hardware that allow multiple virtual guest OS to run simultaneously in a secure and efficient manner. Xen hypervisor is responsible for core hypervisor activities such as CPU, memory virtualization, power management and scheduling of virtual machine [7].

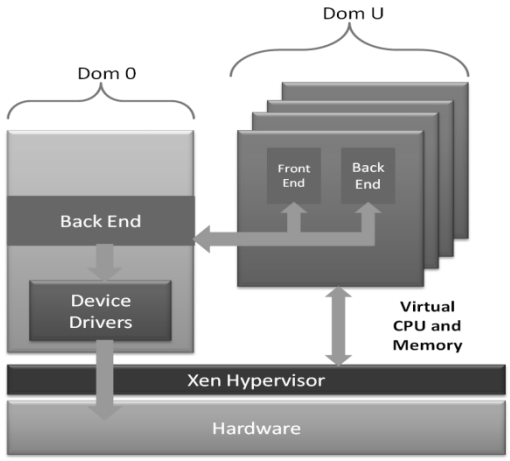


Fig 4: XEN Architecture

Xen architecture has two important and distinct domains: Dom 0 and Dom U. The Dom 0 is a special privilege modified Linux OS. This domain has special rights to access physical I/O resources as well as interact with other virtual machine running on the same physical hardware [5]. It is also use to create and configure all the guest domains known as Dom U, so all Xen virtualization environments require Dom 0 to be running before any other virtual machine.

Dom 0 also contains backend for network and local disk access request from Dom U that contain front end for accessing underlying hardware. This technique create illusion that each frontend have a device dedicated to that domain where as backend acts as a proxy, that understand the details of physical device and send request of each frontend [2].

Dom U's are unprivileged virtual machines that do not have direct access to physical hardware on the machine as Dom 0. Dom U VMs either run with a special modified OS or unmodified OS leveraging special virtualization hardware (Intel or AMD) [10].

D. VMware

The first product of VMware was VMware workstation become successful for desktop virtualization. In late 2000, they have released their first version of server virtualization platform called VMware GSX Server. Later in 2001, they release the first version of Elastic Sky X (ESX) that has different approach to that of VMware workstation.

The core of VMware ESX has three main modules whose primary responsible for regulating CPU affinity, memory allocation and oversubscription, network bandwidth throttling and I/O bandwidth control [2].

The **physical Host Server** is where VMware ESX runs on. VMware had a Hardware Capability List (HCL) that includes the main servers' brand on the market. It is convenient to use ESX on the servers reported on HCL. Another important module of VMware is **VMkernel**. It is a high performance operating system developed by VMware to run on the ESX host server. VMkernel has modular approach, such as allowing new devices to be added without the need of recompiling it. The third important module is **Console Operating System (COS)**. The function of the COS is to provide executing environment for monitoring and administrating ESX.

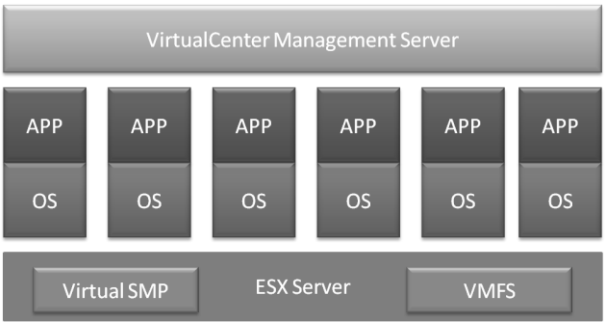


Fig 5: VMware ESX

Along with the above modules, there are two other important components of VMware ESX. One of those is **Virtual Machine File System (VMFS)** that is a high performance cluster file system created by VMware. Another component is **VirtualCenter** that is the management console used to control the virtualized enterprise environment [9].

IV. COMPARISON

In the above section architectural characteristics of each hypervisor product is described. There are other characteristic that is common to each hypervisor that we will see in Table I.

Table I: Comparison of various Hypervisor

Hypervisor	Host OS	Guest OS	Type of Hypervisor
Microsoft Hyper-V	Windows 2008 w/Hyper-v Role, Windows Hyper-V Server	Windows, SUSE	1
KVM	Linux	Linux, Windows	1-2
Xen	Linux, Solaris, NetBSD	BSD, Linux, Solaris, Windos	1
VMware ESX	None-bare-metal	Windows, Linus, Novell, Netware, Sun Solaris, FreeBSD	1

V. CONCLUSION

Hypervisor is a tool that enables one or more operating system to run on the same physical hardware i.e. it enables virtualization. This paper enables researcher to wisely choose correct hypervisor based on the taxonomy proposed on this paper. This paper also provides information to evaluate the existing hypervisor solutions.

ACKNOWLEDGMENT

A special thank to prof. Ritesh patel for giving his guidance and support to carry out this survey and make this survey paper.

REFERENCES

1. F. Belard, "Qemu, a *Fast and Portable Dynamic Translator*," in USENIX Association, 2005. USENIX Annual Technical Conference
2. J, Carlos, C, dos and S, Ramos. *Security Challenges with Virtualization.* 2009.
3. ExpertGlossary at http://www.expertglossary.com/virtualization/definition/hypervisor.
4. **P, Galvin B.** *VMware vSphere Vs. Microsoft Hyper-V: A Technical Analysis.* [White Paper] s.l. : CTI Strategy, 2009.
5. **Xen.** *How Does Xen Works?* [White Paper] 2009.
6. **MSDN.** Hyper-V Architecture. *Microsoft Developer Network.* [Online] MSDN, 2012. http://msdn.microsoft.com/en-us/library/cc768520(v=bts.10).aspx
7. **RedHat.** *KVM – KERNEL BASED VIRTUAL MACHINE.* [White Paper] 2009.
8. **M, Neil.** Hypervisor, Virtualization Stack, And Device Virtualization Architectures. s.l. : Microsoft WINHEC, 2006.
9. **VMware.** *VMware Infrastructure Architecture Overview.* [White Paper] 2006.
10. Xen. *What is Xen Hypervisor?* [White Paper] 2009
11. Zhao X, Borders K & Prakash A Virtual Machine Security Systems