Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING
SKN SINHGAD INSTITUTE OF TECHNOLOGY AND SCIENCE,

LONAVALA-410401

Savitribai Phule Pune University, Pune
2021-22

Case Study No.3

Xen : Paravirtualization, VMware : Full Virtualization, Microsoft Hyper-V

Name : Rathod Akshay Raju

Class :- TE

Division :- B

Roll No. :- 20CETB28

PRN No. :- 72175915h

1. Xen : Paravirtualization :

In computing, **paravirtualization** or **para-virtualization** is a virtualization technique that presents a software interface to the virtual machines which is similar, yet not identical to the underlying hardware–software interface.

The intent of the modified interface is to reduce the portion of the guest's execution time spent performing operations which are substantially more difficult to run in a virtual environment compared to a non-virtualized environment. The paravirtualization provides specially defined 'hooks' to allow the guest(s) and host to request and acknowledge these tasks, which would otherwise be executed in the virtual domain (where execution performance is worse). A successful paravirtualized platform may allow the virtual machine monitor (VMM) to be simpler (by relocating execution of critical tasks from the virtual domain to the host domain), and/or reduce the overall performance degradation of machine execution inside the virtual guest.

Paravirtualization requires the guest operating system to be explicitly ported for the para-API – a conventional OS distribution that is not paravirtualization-aware cannot be run on top of a paravirtualizing VMM. However, even in cases where the operating system cannot be modified, components may be available that enable many of the significant performance advantages of paravirtualization. For example, the Xen Windows GPLPV project provides a kit of paravirtualization-aware device drivers, licensed under the terms of the GPL, that are intended to be installed into a Microsoft Windows virtual guest running on the Xen hypervisor. Such applications tend to be accessible through the paravirtual machine interface environment. This ensures run-mode compatibility across multiple encryption algorithm models, allowing seamless integration within the paravirtual framework.

✞ History :

Paravirtualization is a new term for an old idea. IBM's VM operating system has offered such a facility since 1972 (and earlier as CP-67). In the VM world, this is designated a "DIAGNOSE code", because it uses an instruction code used normally only by hardware maintenance software and thus undefined.

The Parallels Workstation operating system calls its equivalent a "hypercall". All are the same thing: a system call to the hypervisor below. Such calls require support in the "guest" operating system, which has to have hypervisor-specific code to make such calls.

The term "paravirtualization" was first used in the research literature in association with the Denali Virtual Machine Manager.[4] The term is also used to describe the Xen, L4, TRANGO, VMware, Wind River and XtratuM hypervisors. All these projects use or can use paravirtualization techniques to support high performance virtual machines on x86 hardware by implementing a virtual machine that does not implement the hard-to-virtualize parts of the actual x86 instruction set.

A hypervisor provides the virtualization of the underlying computer system. In full virtualization, a guest operating system runs unmodified on a hypervisor. However, improved performance and efficiency is achieved by having the guest operating system communicate with the hypervisor. By allowing the guest operating system to indicate its intent to the hypervisor, each can cooperate to obtain better performance when running in a virtual machine. This type of communication is referred to as paravirtualization.

In 2005, VMware proposed a paravirtualization interface, the Virtual Machine Interface (VMI), as a communication mechanism between the guest operating system and the hypervisor. This interface enabled transparent paravirtualization in which a single binary version of the operating system can run either on native hardware or on a hypervisor in paravirtualized mode. As AMD and Intel CPUs added support for more efficient hardware-assisted virtualization, the standard became obsoleted and VMI support was removed from Linux kernel in 2.6.37 and from VMware products in 2011.

In 2008, Red Hat announced the VirtIO paravirtualization for KVM and Linux, VirtIO driver for Microsoft Windows is also available.

In 2008, Microsoft announced Hyper-V paravirtualization.

✞ Linux Paravirtualization Support :

At the USENIX conference in 2006 in Boston, Massachusetts, a number of Linux development vendors (including IBM, VMware, Xen, and Red Hat) collaborated on an alternative form of paravirtualization, initially developed by the Xen group, called "paravirt-ops". The paravirt-ops code (often shortened to pv-ops) was included in the mainline Linux kernel as of the 2.6.23 version, and provides a hypervisoragnostic interface between the hypervisor and guest kernels. Distribution support for pv-ops guest kernels appeared starting with Ubuntu 7.04 and RedHat 9. Xen hypervisors based on any 2.6.24 or later kernel support pv-ops guests, as does VMware's Workstation product beginning with version 6. VirtualBox also supports it from version 5.0.

2.VMware – Full Virtualization :

In computer science, **virtualization** is a modern technique developed in late 1990s and is different from simulation and emulation. Virtualization employs techniques used to create instances of an environment, as opposed to simulation, which models the environment; or emulation, which replicates the target environment such as certain kinds of virtual machine environments. Full virtualization requires that every salient feature of the hardware be reflected into one of several virtual machines – including the full instruction set, input/output operations, interrupts, memory access, and whatever other elements are used by the software that runs on the bare machine, and that is intended to run in a virtual machine. In such an environment, any software capable of execution on the raw hardware can be run in the virtual machine and, in particular, any operating systems. The obvious test of full virtualization is whether an operating system intended for stand-alone use can successfully run inside a virtual machine.

The cornerstone of full virtualization or type-1 virtualization is a hypervisor or Super Operating system that operates at a higher privilege level than the OS. This Hypervisor or Super OS requires two key features to provision and protect virtualized environments. These two features are:

2. OS-Independent Storage Management to provision resources for all supported Virtual Environments such as Linux, Microsoft Windows or embedded environments and to protect those environments from unauthorized access and,
3. Switching of Virtualized environments to allocate physical computing resources to Virtual Environments.

See Intel VT-x or AMD-V for a detailed description of privilege levels for Hypervisor, OS and User modes, VMCS, VM-Exit and VM-Entry. This virtualization is not to be confused with IBM Virtual Machine implementations of late 60's and early 70's as IBM systems architecture supported only two modes of Supervisor and Program which provided no security or separation of Virtual Machines.

Other forms of platform virtualization allow only certain or modified software to run within a virtual machine. The concept of full virtualization is well established in the literature, but it is not always referred to by this specific term; see platform virtualization for terminology.

An important example of Virtual Machines, not to be confused with Virtualization implemented by emulation was that provided by the control program of IBM's CP/CMS operating system. It was first demonstrated with IBM's CP-40 research system in 1967, then distributed via open source in CP/CMS in 1967–1972, and reimplemented in IBM's VM family from 1972 to the present. Each CP/CMS user was provided a simulated, stand-alone computer. Each such virtual machine had the complete capabilities of the underlying machine, and (for its user) the virtual machine was indistinguishable from a private system. This simulation was comprehensive, and was based on the *Principles of Operation* manual for the hardware. It thus included such elements as an instruction set, main memory, interrupts, exceptions, and device access. The result was a single machine that could be multiplexed among many users.

Full virtualization is possible only with the right combination of hardware and software elements. For example, it was not possible with most of IBM's System/360 series with the exception being the IBM System/360-67; nor was it possible with IBM's early System/370 system. IBM added virtual memory hardware to the System/370 series in 1972 which is not the same as Intel VT-x Rings providing a higher privilege level for Hypervisor to properly control Virtual Machines requiring full access to Supervisor and Program or User modes.

Similarly, full virtualization was not quite possible with the x86 platform until the 2005–2006 addition of the AMD-V and Intel VT-x extensions (see x86 virtualization). Many platform hypervisors for the x86 platform came very close and claimed full virtualization even prior to the AMD-V and Intel VT-x additions. Examples include Adeos, Mac-on-Linux, Parallels Desktop for Mac, Parallels Workstation, VMware Workstation, VMware Server (formerly GSX Server), VirtualBox, Win4BSD, and Win4Lin Pro. VMware, for instance, employs a technique called binary translation to automatically modify x86 software on-thefly to replace instructions that "pierce the virtual machine" with a different, virtual machine safe sequence of instructions; this technique provides the appearance of full virtualization.

A key challenge for full virtualization is the interception and simulation of privileged operations, such as I/O instructions. The effects of every operation performed within a given virtual machine must be kept within that virtual machine – virtual operations cannot be allowed to alter the state of any other virtual machine, the control program, or the hardware. Some machine instructions can be executed directly by the hardware, since their effects are entirely contained within the elements managed by the control program, such as memory locations and arithmetic registers. But other instructions that would "pierce the virtual machine" cannot be allowed to execute directly; they must instead be trapped and simulated. Such instructions either access or affect state information that is outside the virtual machine.

Full virtualization has proven highly successful for:

- sharing a computer system among multiple users;
- isolating users from each other (and from the control program);
- emulating new hardware to achieve improved reliability, security and productivity.

Microsoft Hyper – V :

Microsoft **Hyper-V**, codenamed **Viridian**, and briefly known before its release as **Windows Server Virtualization**, is a native hypervisor; it can create virtual machines on x86-64 systems running Windows. Starting with Windows 8, HyperV superseded Windows Virtual PC as the hardware virtualization component of the client editions of Windows NT. A server computer running Hyper-V can be configured to expose individual virtual machines to one or more networks. Hyper-V was first released with Windows Server 2008, and has been available without additional charge since Windows Server 2012 and Windows 8. A standalone Windows Hyper-V Server is free, but with command-line interface only.

✞ History :

A beta version of Hyper-V was shipped with certain x86-64 editions of Windows Server 2008. The finalized version was released on June 26, 2008 and was delivered through Windows Update. Hyper-V has since been released with every version of Windows Server.

Microsoft provides Hyper-V through two channels:

4. Part of Windows: Hyper-V is an optional component of Windows Server 2008 and later. It is also available in x64 SKUs of Pro and Enterprise editions of Windows 8, Windows 8.1, Windows 10 and Windows 11.
5. Hyper-V Server: It is a freeware edition of Windows Server with limited functionality and Hyper-V component.

Hyper-V Server:

Hyper-V Server 2008 was released on October 1, 2008. It consists of Windows Server 2008 Server Core and Hyper-V role; other Windows Server 2008 roles are disabled, and there are limited Windows services. Hyper-V Server 2008 is limited to a command-line interface used to configure the host OS, physical hardware, and software. A menu driven CLI interface and some freely downloadable script files simplify configuration. In addition, Hyper-V Server supports remote access via Remote Desktop Connection. However, administration and configuration of the host OS and the guest virtual machines is generally done over the network, using either Microsoft Management Consoles on another

Windows computer or System Center Virtual Machine Manager. This allows much easier "point and click" configuration, and monitoring of the Hyper-V Server.

Hyper-V Server 2008 R2 (an edition of Windows Server 2008 R2) was made available in September 2009 and includes Windows PowerShell v2 for greater CLI control. Remote access to Hyper-V Server requires CLI configuration of network interfaces and Windows Firewall. Also using a Windows Vista PC to administer Hyper-V Server 2008 R2 is not fully supported.

✞ Architecture :

Hyper-V implements isolation of virtual machines in terms of a partition. A partition is a logical unit of isolation, supported by the hypervisor, in which each guest operating system executes. There must be at least one parent partition in a hypervisor instance, running a supported version of Windows Server (2008 and later). The virtualization software runs in the parent partition and has direct access to the hardware devices. The parent partition creates child partitions which host the guest OSs. A parent partition creates child partitions using the hypercall API, which is the application programming interface exposed by Hyper-V.

A child partition does not have access to the physical processor, nor does it handle its real interrupts. Instead, it has a virtual view of the processor and runs in Guest Virtual Address, which, depending on the configuration of the hypervisor, might not necessarily be the entire virtual address space. Depending on VM configuration, Hyper-V may expose only a subset of the processors to each partition. The hypervisor handles the interrupts to the processor, and redirects them to the respective partition using a logical Synthetic Interrupt Controller (SynIC). Hyper-V can hardware accelerate the address translation of Guest Virtual Address-spaces by using second level address translation provided by the CPU, referred to as EPT on Intel and RVI (formerly NPT) on AMD.

Child partitions do not have direct access to hardware resources, but instead have a virtual view of the resources, in terms of virtual devices. Any request to the virtual devices is redirected via the VMBus to the devices in the parent partition, which will manage the requests. The VMBus is a logical channel which enables inter-partition communication. The response is also redirected via the VMBus. If the devices in the parent partition are also virtual devices, it will be redirected further until it reaches the parent partition, where it will gain access to the physical devices. Parent partitions run a Virtualization Service Provider (VSP), which connects to the VMBus and handles device access requests from child partitions. Child partition virtual devices internally run a Virtualization Service Client (VSC), which redirect the request to VSPs in the parent partition via the VMBus. This entire process is transparent to the guest OS.

Virtual devices can also take advantage of a Windows Server Virtualization feature, named Enlightened I/O, for storage, networking and graphics subsystems, among others. Enlightened I/O is a specialized virtualization-aware implementation of high level communication protocols, like SCSI, that allows bypassing any device emulation layer and takes advantage of VMBus directly. This makes the communication more efficient, but requires the guest OS to support Enlightened I/O. Currently only the following operating systems support Enlightened I/O, allowing them therefore to run faster as guest operating systems under Hyper-V than other operating systems that need to use slower emulated hardware:

- Windows Server 2008 and later
- Windows Vista and later
- Linux with a 3.4 or later kerne
- FreeBSD

System Requirements :

The Hyper-V role is only available in the x86-64 variants of Standard, Enterprise and Datacenter editions of Windows Server 2008 and later, as well as the Pro, Enterprise and Education editions of Windows 8 and later. On Windows Server, it can be installed regardless of whether the installation is a full or core installation. In addition, Hyper-V can be made available as part of the Hyper-V Server operating system, which is a freeware edition of Windows Server. Either way, the host computer needs the following.

- CPU with the following technologies:
    - NX bit ○ x86-64
    - Hardware-assisted virtualization (Intel VT-x or AMD-V) ○ Second Level Address Translation (in Windows Server 2012 and later)
- At least 2 GB memory, in addition to what is assigned to each guest machine

The amount of memory assigned to virtual machines depends on the operating system:

- Windows Server 2008 Standard supports up to 31 GB of memory for running VMs, plus 1 GB for the host OS.
- Windows Server 2008 R2 Standard supports up to 32 GB, but the Enterprise and Datacenter editions support up to 2 TB. Hyper-V Server 2008 R2 supports up to 1 TB.
- Windows Server 2012 supports up to 4 TB.

The number of CPUs assigned to each virtual machine also depends on the OS:

- Windows Server 2008 and 2008 R2 support 1, 2, or 4 CPUs per VM; the same applies to Hyper-V Server 2008 R2
- Windows Server 2012 supports up to 64 CPUs per VM

There is also a maximum for the number of concurrently active virtual machines.

- Windows Server 2008 and 2008 R2 support 384 per server; Hyper-V Server 2008 supports the same
- Windows Server 2012 supports 1024 per server; the same applies to Hyper-V Server 2012
- Windows Server 2016 supports 8000 per cluster and per node

✞ Limitations :

Audio:

Hyper-V does not virtualize audio hardware. Before Windows 8.1 and Windows Server 2012 R2, it was possible to work around this issue by connecting to the virtual machine with Remote Desktop Connection over a network connection and use its audio redirection feature· Windows 8.1 and Windows Server 2012 R2 add the enhanced session mode which provides redirection without a network connection.

Optical drives pass-through:

Optical drives virtualized in the guest VM are read-only. Officially Hyper-V does not support the host/root operating system's optical drives to pass-through in guest VMs. As a result, burning to discs, audio CDs, video CD/DVD-Video playback are not supported; however, a workaround exists using the iSCSI protocol. Setting up an iSCSI target on the host machine with the optical drive can then be talked to by the standard Microsoft iSCSI initiator. Microsoft produces their own iSCSI Target software or alternative third party products can be used.

VT-x/AMD-V handling:

Hyper-V uses the VT-x on Intel or AMD-V on AMD x86 virtualization. Since Hyper-V is a native hypervisor, as long as it is installed, third-party software cannot use VT-x or AMD-V. For instance, the Intel HAXM Android device emulator (used by Android Studio or Microsoft Visual Studio) cannot run while Hyper-V is installed