WIKIPEDIA

Hardware virtualization

Hardware virtualization is the virtualization of computers as complete hardware platforms, certain logical abstractions of their componentry, or only the functionality required to run various operating systems. Virtualization hides the physical characteristics of a computing platform from the users, presenting instead an abstract computing platform.[1][2] At its origins, the software that controlled virtualization was called a "control program", but the terms "hypervisor" or "virtual machine monitor" became preferred over time.[3]

Contents

Concept

The term "virtualization" was coined in the 1960s to refer to a virtual machine (sometimes called "pseudo machine"), a term which itself dates from the experimental IBM M44/44X system. The creation and management of virtual machines has been called "platform virtualization", or "server virtualization", more recently.

Platform virtualization is performed on a given hardware platform by *host* software (a *control program*), which creates a simulated computer environment, a *virtual machine* (VM), for its *guest* software. The guest software is not limited to user applications; many hosts allow the execution of complete operating systems. The guest software executes as if it were running directly on the physical hardware, with several notable caveats. Access to physical system resources (such as the network access, display, keyboard, and disk storage) is generally managed at a more restrictive level than the *host* processor and system-memory. Guests are often restricted from accessing specific peripheral devices, or may be limited to a subset of the device's native capabilities, depending on the hardware access policy implemented by the virtualization host.

Virtualization often exacts performance penalties, both in resources required to run the hypervisor, and as well as in reduced performance on the virtual machine compared to running native on the physical machine.

Reasons for virtualization

- In the case of server consolidation, many small physical servers are replaced by one larger physical server to decrease the need for more (costly) hardware resources such as CPUs, and hard drives. Although hardware is consolidated in virtual environments, typically OSs are not. Instead, each OS running on a physical server is converted to a distinct OS running inside a virtual machine. Thereby, the large server can "host" many such "guest" virtual machines. This is known as Physical-to-Virtual (P2V) transformation.
- In addition to reducing equipment and labor costs associated with equipment maintenance, consolidating servers can also have the added benefit of reducing energy consumption and the global footprint in environmental-ecological sectors of technology. For example, a typical server runs at 425 W[4] and VMware estimates a hardware reduction ratio of up to 15:1.[5]
- A virtual machine (VM) can be more easily controlled and inspected from a remote site than a physical machine, and the configuration of a VM is more flexible. This is very useful in kernel development and for teaching operating system courses, including running legacy operating systems that do not support modern hardware.[6]
- A new virtual machine can be provisioned as required without the need for an up-front hardware purchase.
- A virtual machine can easily be relocated from one physical machine to another as needed. For example, a salesperson going to a customer can copy a virtual machine with the demonstration software to their laptop, without the need to transport the physical computer. Likewise, an error inside a virtual machine does not harm the host system, so there is no risk of the OS crashing on the laptop.
- Because of this ease of relocation, virtual machines can be readily used in disaster recovery scenarios without concerns with impact of refurbished and faulty energy sources.

However, when multiple VMs are concurrently running on the same physical host, each VM may exhibit varying and unstable performance which highly depends on the workload imposed on the system by other VMs. This issue can be addressed by appropriate installation techniques for temporal isolation among virtual machines.

There are several approaches to platform virtualization.

Examples of virtualization scenarios:

- Running one or more applications that are not supported by the host OS: A virtual machine running the required guest OS could permit the desired applications to run, without altering the host OS.
- Evaluating an alternate operating system: The new OS could be run within a VM, without altering the host OS.
- Server virtualization: Multiple virtual servers could be run on a single physical server, in order to more fully utilize the hardware resources of the physical server.
- Duplicating specific environments: A virtual machine could, depending on the virtualization software used, be duplicated and installed on multiple hosts, or restored to a previously backed-up system state.
- Creating a protected environment: If a guest OS running on a VM becomes damaged in a way that is not cost-effective to repair, such as may occur when studying malware or installing badly behaved software, the VM may simply be discarded without harm to the host system, and a clean copy used upon rebooting the guest .

Full virtualization

In full virtualization, the virtual machine simulates enough hardware to allow an unmodified "guest" OS designed for the same instruction set to be run in isolation. This approach was pioneered in 1966 with the IBM CP-40 and CP-67, predecessors of the VM family.

Hardware-assisted virtualization

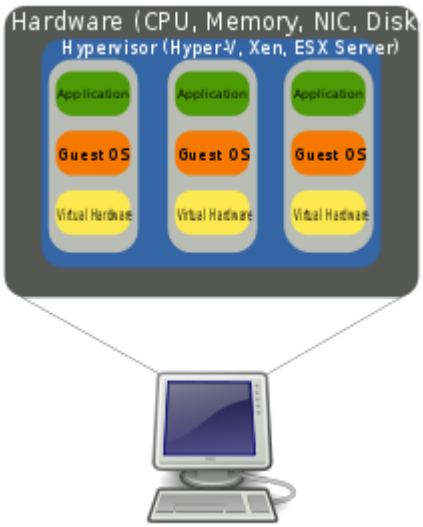
In hardware-assisted virtualization, the hardware provides architectural support that facilitates building a virtual machine monitor and allows guest OSes to be run in isolation.[7] Hardware-assisted virtualization was first introduced on the IBM System/370 in 1972, for use with VM/370, the first virtual machine operating system.

In 2005 and 2006, Intel and AMD provided additional hardware to support virtualization. Sun Microsystems (now Oracle Corporation) added similar features in their UltraSPARC T-Series processors in 2005.

In 2006, first-generation 32- and 64-bit x86 hardware support was found to rarely offer performance advantages over software virtualization.[8]

Paravirtualization

In paravirtualization, the virtual machine does not necessarily simulate hardware, but instead (or in addition) offers a special API that can only be used by modifying the "guest" OS. For this to be possible, the "guest" OS's source code must be available. If the source code is available, it is sufficient to replace sensitive instructions with calls to VMM APIs (e.g.: "cli" with "vm_handle_cli()"), then re-compile the OS and use the new binaries. This system call to the hypervisor is called a "hypercall" in TRANGO and Xen; it is implemented via a DIAG ("diagnose") hardware instruction in IBM's CMS under VM (which was the origin of the term *hypervisor*)..



Logical diagram of full virtualization.

Operating-system-level virtualization

In operating-system-level virtualization, a physical server is virtualized at the operating system level, enabling multiple isolated and secure virtualized servers to run on a single physical server. The "guest" operating system environments share the same running instance of the operating system as the host system. Thus, the same operating system kernel is also used to implement the "guest" environments, and applications running in a given "guest" environment view it as a stand-alone system.

Hardware virtualization disaster recovery

A disaster recovery (DR) plan is often considered good practice for a hardware virtualization platform. DR of a virtualization environment can ensure high rate of availability during a wide range of situations that disrupt normal business operations. In situations where continued operations of hardware virtualization platforms is important, a disaster recovery plan can ensure hardware performance and maintenance requirements are met. A hardware virtualization disaster recovery plan involves both hardware and software protection by various methods, including those described below.[9][10]

Tape backup for software data long-term archival needs
: This common method can be used to store data offsite, but data recovery can be a difficult and lengthy process. Tape backup data is only as good as the latest copy stored. Tape backup methods will require a backup device and ongoing storage material.

Whole-file and application replication
: The implementation of this method will require control software and storage capacity for application and data file storage replication typically on the same site. The data is replicated on a different disk partition or separate disk device and can be a scheduled activity for most servers and is implemented more for database-type applications.

Hardware and software redundancy
: This method ensures the highest level of disaster recovery protection for a hardware virtualization solution, by providing duplicate hardware and software replication in two distinct geographic areas.[11]

See also

- Application virtualization

- Comparison of platform virtualization software
- Desktop virtualization
- Dynamic infrastructure
- Hyperjacking
- Instruction set simulator
- Popek and Goldberg virtualization requirements
- Physicalization
- Virtual appliance
- Virtualization for aggregation
- Workspace virtualization

References

1. Turban, E; King, D.; Lee, J.; Viehland, D. (2008). "19". *Electronic Commerce A Managerial Perspective* (http://wps.prenhall.com/wps/media/objects/5073/5195381/pdf/Online_Chapter_19.pdf) (PDF) (5th ed.). Prentice-Hall. p. 27.
2. "Virtualization in education" (http://www-07.ibm.com/solutions/in/education/download/Virtualization%20in%20Education.pdf) (PDF). IBM. October 2007. Retrieved 6 July 2010.
3. Creasy, R.J. (1981). "The Origin of the VM/370 Time-sharing System" (http://pages.cs.wisc.edu/~stjones/proj/vm_reading/ibmrd2505M.pdf) (PDF). IBM. Retrieved 26 February 2013.
4. [1] (http://msdn.microsoft.com/en-us/library/dd393312.aspx) Profiling Energy Usage for Efficient Consumption; Rajesh Chheda, Dan Shookowsky, Steve Stefanovich, and Joe Toscano
5. VMware server consolidation overview (http://www.vmware.com/solutions/consolidation/)
6. Examining VMware (http://systems.cs.columbia.edu/archive/pub/2000/08/examining-vmware/) Dr. Dobb's Journal August 2000 By Jason Nieh and Ozgur Can Leonard
7. Uhlig, R. et al.; "Intel virtualization technology," Computer , vol.38, no.5, pp. 48-56, May 2005
8. A Comparison of Software and Hardware Techniques for x86 Virtualization, Keith Adams and Ole Agesen, VMware, ASPLOS'06 21–25 October 2006, San Jose, California, USA (http://www.vmware.com/pdf/asplos235_adams.pdf) "Surprisingly, we find that the first-generation hardware support rarely offers performance advantages over existing software techniques. We ascribe this situation to high VMM/guest transition costs and a rigid programming model that leaves little room for software flexibility in managing either the frequency or cost of these transitions."
9. "The One Essential Guide to Disaster Recovery: How to Ensure IT and Business Continuity" (https://web.archive.org/web/20110516030950/http://www.visionsolutions.com/downloads/whitepapers/EssentialDR.pdf) (PDF). Vision Solutions, Inc. 2010. Archived from the original (http://www.visionsolutions.com/Downloads/Whitepapers/EssentialDR.pdf) (PDF) on 16 May 2011.
10. Wold, G (2008). "Disaster Recovery Planning Process" (https://web.archive.org/web/20120815054111/http://www.drj.com/new2dr/w2_002.htm). Archived from the original (http://www.drj.com/new2dr/w2_002.htm) on 15 August 2012.
11. "Disaster Recovery Virtualization Protecting Production Systems Using VMware Virtual Infrastructure and Double-Take" (https://web.archive.org/web/20100923021435/http://www.vmware.com/files/pdf/DR_VMware_DoubleTake.pdf) (PDF). VMware. 2010. Archived from the original (http://www.vmware.com/files/pdf/DR_VMware_DoubleTake.pdf) (PDF) on 23 September 2010.

External links

- An introduction to Virtualization (http://www.kernelthread.com/publications/virtualization/)
- Xen and the Art of Virtualization (http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf), ACM, 2003, by a group of authors
- Linux Virtualization Software (https://web.archive.org/web/20070209020711/http://www-128.ibm.com/developerworks/library/l-linuxvirt/index.html)
- Zoppis, Bruno (27 August 2007) [1st pub. LinuxDevices:2007]. "Using a hypervisor to reconcile GPL and proprietary embedded code" (https://web.archive.org/web/20131224184029/http://archive.linuxgizmos.com/using-a-hypervisor-to-

reconcile-gpl-and-proprietary-embedded-code/). *Linux Gizmos*. Archived from the original (http://archive.linuxgizmos.com/using-a-hypervisor-to-reconcile-gpl-and-proprietary-embedded-code/) on 24 December 2013.
