Virtualization is the process of running a virtual instance of a computer system in a layer abstracted from the actual hardware. Most commonly, it refers to running multiple operating systems on a computer system simultaneously. To the applications running on top of the virtualized machine, it can appear as if they are on their own dedicated machine, where the operating system, libraries, and other programs are unique to the guest virtualized system and unconnected to the host operating system which sits below it.

There are many reasons why people utilize virtualization in computing. To desktop users, the most common use is to be able to run applications meant for a different operating system without having to switch computers or reboot into a different system. For administrators of servers, virtualization also offers the ability to run different operating systems, but perhaps, more importantly, it offers a way to segment a large system into many smaller parts, allowing the server to be used more efficiently by several different users or applications with different needs. It also allows for isolation, keeping programs running inside of a virtual machine safe from the processes taking place in another virtual machine on the same host.

Some terminologies associated with Virtualization

* Hypervisor: It is an operating system, performing on the actual hardware, the virtual counterpart is a subpart of this operating system in the form of a running process. Hypervisors are observed as Domain 0 or Dom0.
* Virtual Machine (VM): It is a virtual computer, executing underneath a hypervisor.
* Container: Some light weighted VMs that are subpart of the same operating system instance as its hypervisor are known as containers. They are a group of processes that runs along with their corresponding namespace for process identifiers.
* Virtualization Software: Either be a piece of a software application package or an operating system or a specific version of that operating system, this is the software that assists in deploying the virtualization on any computer device.
* Virtual Network: It is a logically separated network inside the servers that could be expanded across multiple servers.

Characteristics of Virtualization

1. Resource Distribution: Either be a single computer or a network of connected servers, virtualization allows users to make a unique computer environment from one host machine that lets users to restrict the participants as active users, scale down power consumption and easy control.
2. Isolation: Virtualization software involves self-contained virtual machines, these VMs give guest users (not an individual but several instances as applications, operating systems, and devices) an isolated online, virtual environment. This online environment not only defends sensitive knowledge but also allows guest users to remain-connected.
3. Availability: Virtualization software provides various number of features that users won’t obtain at physical servers, these features are beneficial in increasing uptime, availability, fault tolerance, and many more. These features help users to avoid downtime that subverts the users’ efficiencies and productivities and generates security threats and safety hazards.
4. Aggregation: Since virtualization allows several devices to split resources from a single machine, so it can be deployed to join multiple devices into a single potent host. In addition to that, aggregation also demands for cluster management software to connect a homogeneous group of computers or servers collectively for making a unified resource centre.
5. Authenticity and security: At ease, virtualization platforms assure the continuous uptime by balancing load automatically that runs an excessive number of servers across multiple host machines to prevent interruption services.

* Full virtualization

Full virtualization is a common and cost-effective type of virtualization, which is basically a method by which computer service requests are separated from the physical hardware that facilitates them. With full virtualization, operating systems and their hosted software are run on top of virtual hardware. It differs from other forms of virtualization (like paravirtualization and hardware-assisted virtualization) in its total isolation of guest operating systems from their hosts.

Full Virtualization provides a complete simulation of the underlying hardware allowing execution of unmodified operating systems in the virtual machines. It requires that every salient feature of the hardware be reflected into every one of several virtual machines. In Full Virtualization machine language code of the guest OS is converted into the machine language code of the host through a binary translation process.

Diagram

Description automatically generated

* Paravirtualization

Paravirtualization (PV) is an enhancement of virtualization technology in which a guest operating system (guest OS) is modified prior to installation inside a virtual machine (VM) in order to allow all guest OS within the system to share resources and successfully collaborate, rather than attempt to emulate an entire hardware environment.

With paravirtualization, virtual machines can be accessed through interfaces that are similar to the underlying hardware. This capacity minimizes overhead and optimizes system performance by supporting the use of VMs that would otherwise be underutilized in conventional or full hardware virtualization.

The main limitation of PV is the fact that the guest OS must be tailored specifically to run on top of the virtual machine monitor (VMM), the host program that allows a single computer to support multiple, identical execution environments. However, paravirtualization eliminates the need for the virtual machine to trap privileged instructions. Trapping, a means of handling unexpected or unallowable conditions, can be time-consuming and can adversely impact performance in systems that employ full virtualization.

Paravirtualization attempts to resolve issues found in full virtualization. The primary difference between paravirtualization and full virtualization is the ability to make modifications to the guest OS in PV. Furthermore, in PV, the guest OS is aware it is being virtualized. In full virtualization, the unmodified OS is unaware it its being virtualized and sensitive OS calls are captured and translated using binary translation.

In PV, the guest kernel is modified to run with the hypervisor. This frequently involves removing operations that only run-in ring 0 of the processor with calls to the hypervisor, or hypercalls.

Paravirtualization offers various performance advantages as well as efficiencies that offer improved scaling. It is useful in a variety of technical fields, including:

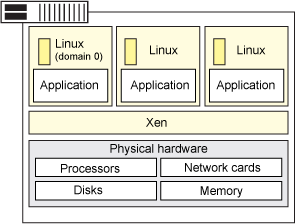
* disaster recovery.
* capacity management.
* separating test systems and development environments; and
* transferring data from one system to another.
* Partial virtualization

Partial virtualization provides a partial emulation of the underlying hardware, thus not allowing the complete execution of the guest operating system in complete isolation. Partial virtualization allows many applications to run transparently, but not all the features of the operating system can be supported, as happens with full virtualization. An example of partial virtualization is address space virtualization used in time-sharing systems.



Xen is a type 1 hypervisor that creates logical pools of system resources so that many virtual machines can share the same physical resources.

Xen is a hypervisor that runs directly on the system hardware. Xen inserts a virtualization layer between the system hardware and the virtual machines, turning the system hardware into a pool of logical computing resources that Xen can dynamically allocate to any guest operating system. The operating systems running in virtual machines interact with the virtual resources as if they were physical resources.



Xen is primarily a bare-metal, type-1 hypervisor that can be directly installed on computer hardware without the need for a host operating system. Because it's a type-1 hypervisor, Xen controls, monitors, and manages the hardware, peripheral and I/O resources directly. Guest virtual machines request Xen to provision any resource and must install Xen virtual device drivers to access hardware components. Xen supports multiple instances of the same or different operating systems with native support for most operating systems, including Windows and Linux. Moreover, Xen can be used on x86, IA-32 and ARM processor architecture.

## **Features**

### Full virtualization

Most hypervisors are based on full virtualization which means that they completely emulate all hardware devices to the virtual machines. Guest operating systems do not require any modification and behave as if they each have exclusive access to the entire system.

Full virtualization often includes performance drawbacks because complete emulation usually demands more processing resources (and more overhead) from the hypervisor. Xen is based on paravirtualization; it requires that the guest operating systems be modified to support the Xen operating environment. However, the user space applications and libraries do not require modification.

Operating system modifications are necessary for reasons like:

* So that Xen can replace the operating system as the most privileged software.
* So that Xen can use more efficient interfaces (such as virtual block devices and virtual network interfaces) to emulate devices — this increases performance.

### Xen can run multiple guest OS each in its on VM

Xen can run several guest operating systems each running in its own virtual machine or domain. When Xen is first installed, it automatically creates the first domain, Domain 0 (or dom0).

### Instead of a driver, lots of great stuff happens in the Xen daemon

The Xen daemon, xend, is a Python program that runs in dom0. It is the central point of control for managing virtual resources across all the virtual machines running on the Xen hypervisor. Most of the command parsing, validation, and sequencing happens in user space in xend and not in a driver.

## **Choosing Xen**

On the pro side:

* The Xen server is built on the open-source Xen hypervisor and uses a combination of paravirtualization and hardware-assisted virtualization. This collaboration between the OS and the virtualization platform enables the development of a simpler hypervisor that delivers highly optimized performance.
* Xen provides sophisticated workload balancing that captures CPU, memory, disk I/O, and network I/O data; it offers two optimization modes: one for performance and another for density.
* The Xen server takes advantage of a unique storage integration feature called the Citrix Storage Link. With it, the sysadmin can directly leverage features of arrays from such companies as HP, Dell Equal Logic, NetApp, EMC, and others.

On the con side:

* Xen has a relatively large footprint and relies on Linux in dom0.
* Xen relies on third-party solutions for hardware device drivers, storage, backup and recovery, and fault tolerance.
* Xen gets bogged down with anything with a high I/O rate or anything that sucks up resources and starves other VMs.
* Xen’s integration can be problematic; it could become a burden on your Linux kernel over time.