



# International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: [www.ijarcsse.com](http://www.ijarcsse.com)

## Different Ways to Define Virtualization

Archana\*

CSE Department SBIET,  
Pundri, Haryana, India

Neeraj Gandhi

CSE Department SKIET,  
Kurukshetra, Haryana, India

**Abstract**— *Virtualization presents IT's most impressive challenge: the infrastructure cover that compels IT departments to channel 75 percent of their budget into maintenance, leaving limited resources for business-building innovation. The difficulty stems from the architecture of today's X86 computers: they are designed to run one operating system and application at a time. As a result, even small data centers have to install many servers, each operating at just five percent to 15 percent of capacity which is highly inefficient by all standards. Virtualization software solves the problem by enabling a number of operating systems and applications to run on one physical server or host. Each self-contained virtual machine is isolated from the others, and uses as much of the host's computing resources as it requires.*

**Keywords**— *Virtualization, Virtualization Types, Server Virtualization, Hardware Virtualization, Virtualized vs traditional*

### I. INTRODUCTION

Virtualization is defined as the abstraction of objects, creating a virtual version of objects such as a server or storage device. For example, when you partition a hard drive into two partitions - C and D, say - you create virtual drives but the physical hard drive has not changed. Similarly, Network attached storage (NAS) presents users with an abstract storage space - a single point of access to data - though in reality NAS contains many drives and tapes hidden from the end user. In virtualization, a software layer is inserted either directly on the physical hardware or on the host operating system. This software (also known as hypervisor or virtual machine monitor) allocates hardware resources (CPU, RAM, storage and networking) and allows a single server to run multiple operating system images at the same time. Each one of these operating systems is a `virtual server. Virtual machines are completely compatible with all standards x86 operating systems, applications and device drivers. Virtualization allows higher utilization of servers, which usually run at a fraction of their capacity. By replacing physical servers with virtual ones, and consolidating many together, it is possible to reduce datacenter space, energy consumption, hardware costs and maintenance personnel. It is also easier to move virtual servers between hosts to form high availability, fault tolerant or other virtual clusters, thus improving business continuity, performance and responsiveness.

### II. EVOLUTION OF VIRTUALIZATION

The idea of virtualization has been around for many years. It was first used in the 60's as a way to partition large mainframe hardware. Engineers then faced the same problems of today, such as too many underutilized servers. IBM breaks new ground of virtualization by allowing engineers to partition mainframes to allow tasks to multi task. Server virtualization on the x86 platforms was invented in the 90's primarily by VMware. Since then many companies have entered into the x86 hardware and software virtualization market including Microsoft, Citrix and more recently Sun with VirtualBox and Solaris zones. Sun also has been experimenting with hardware virtualization on their SPARC platform. The need to virtualize is ever growing due to the growing demand and lessening availability of data center space.

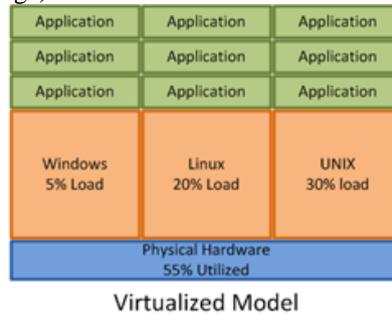
### III. TRADITIONAL vs VIRTUALIZED

An environment is called traditional if there is no virtualization applied to it. A traditional environment can be described as "one server, one OS, one application" -concept. When a new application is introduced, server hardware is ordered and after its arrival, the OS is installed. After the OS installation the application itself is installed and configured. At this point the system is ready for production use, and normal maintenance tasks are started (e.g. monitoring hardware and applications, creating backups).

Application	Application	Application
Application	Application	Application
Application	Application	Application
Windows 5% Load	Linux 20% Load	UNIX 30% Load
Physical Hardware 5% Utilized	Physical Hardware 20% Utilized	Physical Hardware 30% Utilized

Traditional Model

Virtualization is defined as the capability to run more than one application or operating system in an isolated and secure setting on a single physical system. As this figure show, the basic concept of virtualization is to run more than one operating system and application on the same physical hardware. For most organizations most servers are only utilized between 7% and 16% on average. That means that if there are 100 servers sitting in a datacenter that they could be consolidated to between 7 and 16 (on average).



That means for fewer servers to power and maintain, which can show huge cost savings over time. Using this technology also makes IT infrastructure easier to manage and far more flexible which will lead to less downtime. Now for some terms:

- **Host:** Host is defined as a physical machine that is hosting virtual machines
- **Virtual Machine:** Virtual Machine is defined as the software container and supporting files that will contain operations of the hosted software
- **Hypervisor:** Hypervisor is defined as a software or firmware that provides a means for a virtual machine to interact directly with the underlying physical hardware without emulation but still within specified constraints. It is also referred to as bare metal virtualization.
- **I/O:** I/O stands for Input/Output and is defined as any type of input or output coming from or going to a piece of software.
- **Memory Page:** Memory Page is a term for a block in memory where information is stored.

#### IV. DIFFERENT WAYS TO DEFINED VIRTUALIZATION

Virtualization is mainly of three categories: Operating System, Storage, and Applications. But these categories are very broad and don't sufficiently define the key aspects of data center virtualization. It's helpful to refine these categories into eight, specific categories to carefully understand the similarities and differences between the definitions of virtualization.

##### OPERATING SYSTEM VIRTUALIZATION

The most common form of virtualization today is operating systems virtualization. Operating systems is quickly becoming a core component of the IT Infrastructure. Generally, it is the form of virtualization to which end-users are most familiar with. Virtual machines are normally full implementations of standard operating systems, such as Windows or RedHat Enterprise Linux, running simultaneously on the same physical hardware. Virtual Machine Managers manage each virtual machine individually; each OS instance is unaware that it's virtual and other virtual operating systems are running at the same time. Companies like Microsoft, VMware, Intel, and AMD are leading the way in breaking the physical relationship between an operating system and its local hardware, extending this model into the data center. This trend ultimately saves enterprises money on hardware, co-location fees, rack space, power, cable management, and more.

##### APPLICATION SERVER VIRTUALIZATION

The core concept of application server virtualization is best seen with a reverse proxy load balancer: a service that provides access to lots of different application services transparently. In a typical deployment, a reverse proxy will host a virtual interface accessible to the end user on the front end. On the back end, the reverse proxy will load balance a number of different servers and applications such as a web server. The virtual interface referred to as a Virtual IP or VIP. Virtual IP represents itself as the actual web server, and manages the connections to and from the web server as needed. This enables the load balancer to manage multiple web servers or applications as a single instance that provides a more secure and robust topology than one allowing users direct access to individual web servers. Application Server Virtualization can be applied to any types of application deployments and architectures, from fronting application logic servers to distributing the load between multiple web server platforms.

##### APPLICATION VIRTUALIZATION

Application Server and Application Virtualization are two completely different concepts. Thin clients are referred as application virtualization. The technology is exactly the same, only the name has changed to make it more IT-PC (politically correct). Soft grid by Microsoft is an outstanding example of deploying application virtualization. Although you may be running Microsoft Word 2007 locally on your laptop, the binaries, personal information, and running state are all stored on, managed, and delivered by Soft grid. Your local laptop provides the CPU and RAM required running the software, but nothing is installed locally on your own machine. Other types of Application Virtualization include

Microsoft Terminal Services and browser-based applications. All of these implementations depend on the virtual application running locally and the management and application logic running remotely.

### **MANAGEMENT VIRTUALIZATION**

Chances are you already implement administrative virtualization throughout your IT organization, but you probably don't refer to it by this phrase. If you implement separate passwords for your root/administrator accounts between your mail and web servers, and your mail administrators don't know the password to the web server and vice versa, then you've deployed management virtualization in its most basic form. The prototype can be extended down to segmented administration roles on one platform or box, which is where segmented administration becomes "virtual." User and group policies in Microsoft Windows XP, 2003, and Vista are an excellent example of virtualized administration rights. Management virtualization is also a key concept in overall data center management. It is critical that the network administrators have full access to all the infrastructure gear, such as core routers and switches, but that they not have admin-level access to servers

### **NETWORK VIRTUALIZATION**

Network virtualization may be the most confusing, definition of virtualization. For brevity, the scale of this discussion is relegated to what amounts to virtual IP management and segmentation. A simple example of IP virtualization is a VLAN. A single Ethernet port may support multiple virtual connections from multiple IP addresses and networks, but they are virtually segmented using VLAN tags. Each virtual IP connection over this single physical port is independent and unaware of others' existence, but the switch is aware of each unique connection and manages each one independently. Another example is virtual routing tables. A routing table and an IP network port share a 1:1 relationship, even though that single port may host multiple virtual interfaces. The single routing table will contain multiple routes for each virtual connection, but they are still managed in a single table. Virtual routing tables change that example into a one to many relationship, where any single physical interface can maintain multiple routing tables, each with multiple entries. This provides the interface with the ability to bring up (and tear down) routing services on the fly for one network without interrupting other services and routing tables on that same interface.

### **HARDWARE VIRTUALIZATION**

Hardware virtualization is very similar in concept to OS/Platform virtualization, and to some degree is required for OS virtualization to occur. Hardware virtualization breaks up pieces and locations of physical hardware into independent segments and manages those segments as separate, individual components. Although they fall into different classifications, both symmetric and asymmetric multiprocessing are examples of hardware virtualization. In both instances, the process requesting CPU time isn't aware which processor it's going to run on; it just requests CPU time from the OS scheduler and the scheduler takes the responsibility of allocating processor time. As far as the process is concerned, it could be spread across any number of CPUs and any part of RAM, so long as it's able to run unaffected. Another example of hardware virtualization is "slicing": carving out precise portions of the system to run in a "walled garden," such as allocating a fixed 25% of CPU resources to bulk encryption. If there are no processes that need to crunch numbers on the CPU for block encryption, then that 25% of the CPU will go unutilized. If too many processes need mathematical computations at once and require more than 25%, they will be queued and run as a FIFO buffer because the CPU isn't allowed to give out more than 25% of its resources to encryption. This type of hardware virtualization is sometimes referred to as pre-allocation. Asymmetric multiprocessing is a form of pre-allocation virtualization where certain tasks are only run on certain CPUs. In contrast, symmetric multiprocessing is a form of dynamic allocation, where CPUs are interchangeable and used as needed by any part of the management system. Each classification of hardware virtualization is unique and has value, depending on the implementation. Pre-allocation virtualization is perfect for very specific hardware tasks, such as offloading functions to a highly optimized, single-purpose chip. However, pre-allocation of commodity hardware can cause artificial resource shortages if the allocated chunk is underutilized. Dynamic allocation virtualization is a more standard approach and typically offers greater benefit when compared to pre-allocation. For true virtual service provisioning, dynamic resource allocation is important because it allows complete hardware management and control for resources as needed; virtual resources can be allocated as long as hardware resources are still available. The downside to dynamic allocation implementations is that they typically do not provide full control over the dynamicity, leading to processes which can consume all available resources.

### **STORAGE VIRTUALIZATION**

As another example of a tried-and-true technology that's been dubbed "virtualization," storage virtualization can be broken up into two general classes: block virtualization and file virtualization. Block virtualization is best summed up by Storage Area Network (SAN) and Network Attached Storage (NAS) technologies: distributed storage networks that appear to be single physical devices. Under the hood, SAN devices themselves typically implement another form of Storage Virtualization: RAID. iSCSI is another very common and specific virtual implementation of block virtualization, allowing an operating system or application to map a virtual block device, such as a mounted drive, to a local network adapter (software or hardware) instead of a physical drive controller. The iSCSI network adapter translates block calls from the application to network packets the SAN understands and then back again, essentially providing a virtual hard drive. File virtualization moves the virtual layer up into the more human-consumable file and directory structure level. Most file virtualization technologies sit in front of storage networks and keep track of which files and directories reside

on which storage devices, maintaining global mappings of file locations. When a request is made to read a file, the user may think this file is statically located on their personal remote drive, P:\My Files\budget.xls; however, the file virtualization appliance knows that the file is actually located on an SMB server in a data center across the globe at //10.0.16.125/finance/alice/budget-document /budget.xls. File-level virtualization obfuscates the static virtual location pointer of a file (in this case, on Alice's P:\ drive) from the physical location, allowing the back-end network to remain dynamic. If the IP address for the SMB server has to change, or the connection needs to be re-routed to another data center entirely, only the virtual appliance's location map needs to be updated, not every user that needs to access their P:\ drive.

## **SERVICE VIRTUALIZATION**

And finally, we reach the macro definition of virtualization: service virtualization. Service virtualization is consolidation of all of the above definitions into one catch-all catchphrase. Service virtualization connects all of the components utilized in delivering an application over the network, and includes the process of making all pieces of an application work together regardless of where those pieces physically reside. This is why service virtualization is typically used as an enabler for application availability. For example, a web application typically has many parts: the user-facing HTML; the application server that processes user input; the SOA gears that coordinate service and data availability between each component; the database back-end for user, application, and SOA data; the network that delivers the application components; and the storage network that stores the application code and data. Service virtualization allows each one of the pieces to function independently and be "called up" as needed for the entire application to function properly. When we look deeper into these individual application components, we may see that the web server is load-balanced between 15 virtual machine operating systems, the SOA requests are pushed through any number of XML gateways on the wire, the database servers may be located in one of five global data centers, and so on. Service virtualization combines these independent pieces and presents them together to the user as a single, complete application. While Service virtualization may encompass all the current definitions of virtualization, it's by no means where IT will stop defining the term. With the pervasive and varied use of the word (as well as the technologies it refers to), there may never be a "final" definition for virtualization; it will continue to evolve and expand as more and more technologies become less and less dependent on rigid operating environments.

## **V. CONCLUSIONS**

Though virtualization improves system utilization, security and robustness, possible liabilities that surface as we attempt to leverage virtualization for software recomposition include: (1) scalability in terms of dynamic system properties, (2) sustainability in terms of software engineering challenges of constructing service composition. Virtualization falls into three categories: Operating System, Storage, and Applications. But these categories are very broad and don't adequately delineate the key aspects of data center virtualization. It's helpful to distill these broader categories into eight, specific categories to thoroughly understand the differences (and similarities) between the definitions of virtualization.

## **REFERENCES**

- [1] Gabriel Torres, "Everything You Need To Know About the Intel Virtualization Technology [Http://www.hardwaresecrets.com/printpage/Everything-You-Need-To-Know-About-The-Intel-Virtualization-Technology/263](http://www.hardwaresecrets.com/printpage/Everything-You-Need-To-Know-About-The-Intel-Virtualization-Technology/263), July 2012
- [2] Cpt. OGÎGĂU NEAMȚIU FLORIN, "Improving IT Services Through Virtualization " The 6<sup>th</sup> International Scientific Conference DEFENSE RESOURCES MANAGEMENT IN THE 21st CENTURY , December 02-03, 2011 , PP 165-170
- [3] Abhijit Shroff and Venkata Reddy Donthireddy, "Virtualization Imperatives and Performance", Setlabs Briefings, Vol. 7.1, Pp 97-102, 2011
- [4] Virtualization Special Interest Group PCI Security Standards Council "Information Supplement, PCI DSS Virtualization Guidelines "June 2011
- [5] Kirk L. Kroeker "The Evolution of Virtualization" Communication of the ACM, March 2009, Vol. 52.3, pp 18-20
- [6] Kirill Kolyshkin, "Virtualization in Linux", Sep. 1, 2006, pp 1-5
- [7] Alan Murphy, "Virtualization Defined- Eight Different Ways", F5 Technical Marketing Manager, Security, White Paper, pp 1-3
- [8] Intel Corporation. IA-32 Intel Architecture Software Developer's Manual, Volume 1: Basic Architecture, 2003.
- [9] VMware, Inc. VMware ESX Server. User's Manual. Version 1.5. 2002.
- [10] Kozierok and Charles M," The PC Guide – System Resources", April 17, 2001