VMware® Infrastructure 3

Advanced Technical Design Guide

~and~

Advanced Operations Guide

Two books in one!

Ron Oglesby
Scott Herold
Mike Laverick

Chapter 1 – Virtualization Overview

This book is an updated version of the original *ESX Server Advanced Technical Design Guide* that was written for ESX 2.x. It has been updated, added to, and re-written to bring it up to speed for ESX 3.0 and Virtual Center 2.0. As was the case with the original book, this book is designed to provide you with practical, real-world information about the use and design of VMware ESX Server systems. In order to study the advanced technical details, we need to ensure that you have a good baseline understanding of VMware ESX features, VMware's other products, and of course virtualization concepts as we know them today.

Even if you've already read and understand our previous book or if you're very familiar with previous versions of ESX or VMware server, we still recommend that you at least skim through this chapter so that you can familiarize yourself with the specific terms and phrasing that we'll use throughout this book.

As you read through this book, you should keep in mind that each chapter was written separately and that it's okay for you to read them out-of-order. Of course if you're new to VMware and virtualization technologies, the chapter sequence will guide you through a logical progression of the components. We'll begin in this chapter with an overview of VMware software solutions.

Virtualization - Today's Favorite Buzz Word

Standing at the RapidApp booth at VMworld this year (VMware's annual conference), I had one of the attendee's approach me and ask "so do you guys really do something with virtualization technology, or are you just another company that throws the word virtual in front of the product and come here?" I laughed. It was funny and any of you that have watched virtualization grow over the past few years understands how funny that really is. Virtualization is the hottest topic around the IT space right now. Virtualize your servers, virtualize your network, virtualize your storage, virtualize your boss… well that last one isn't here yet, but we're working on it.

This book deals with the market leading product ESX Server 3. In this chapter we will give you an overview of where ESX (or as VMware marketing people like to call the whole solution "Virtual Infrastructure 3") fits into the server virtualization space, the type of virtualization it does, and we will try to keep away from all the other "virtual" products.

VMware ESX Server allows for the "virtualization" of x86-based servers. The basic concept is that a single piece of physical hardware is used to host multiple logical or "virtual" servers. The ability to host multiple virtual servers on a single piece of hardware, while simple to understand in theory, can be extremely complex in execution. It should be noted that the idea of virtual servers is nothing new. Look to the UNIX space and you will hear terms like "Frames" and LPARs" and can equate those with Hosts and Virtual Machines. VMware's place in IT shops is driven in the x86 space, where hardware is cheap (compared to UNIX) and generally underutilized.

When VMware technology is used in an environment, a single host allows multiple virtual servers to share the host's physical resources. These resources include processor, memory, network cards, and storage (We'll refer to these four physical resources as the "core four" throughout this book.) This architecture also gives you the ability to "partition" your host to allow for the server's resources to be fully utilized while you migrate multiple physical servers to a single VMware server running multiple virtual servers. (We'll discuss partitioning and migration strategies in much more depth throughout this book.)

So what does this mean to your environment or your business? It depends on what your business looks like and which product from VMware you're thinking about using. Obviously if you're reading this book then you're at least contemplating an ESX Server implementation. But even with ESX you have a number of alternatives on how to deploy and utilize the system. Also, when investigating VMware technology it is unfair to simply look at ESX and why it is used. You can only decide to use ESX instead of GSX or Workstation after you really understand what each product does and where they truly fit. Outside of the VMware suite of products you may also have to look at other virtualization solutions, like those based on XenSource. And to make an informed decision about these solutions you need a better understanding of virtualization technology for x86 servers.

Hardware Virtualization vs. Software

The word virtualization is thrown around a lot these days, and everyone claims his way of virtualizing is better than everyone else's. But to understand what you are getting into you need to understand some basics. The first (and most important) concept you need to grasp is the difference between hardware virtualization and software. Most people in IT are aware that Intel and AMD have released "processor" virtualization technologies for their processor. There is a lot of talk about "moving this or that out of Ring 0 and into a Ring 1." There is also talk about how Intel is working on hardware hypervisors for their network cards, and HBA manufacturers are also creating hypervisors… So what does it all mean?

To make it simple let's look at the processor and two major virtualization "products." First let's look at XenSource (pronounced zen) and Xen based products like Virtual Iron. Xen is basically a free virtual machine monitor (a chunk of software that allows for multiple operating systems to run on a single piece of hardware) that grew out of the Linux community and originally would require a modification of the guest operating system to run properly. There is more detail and strategy on this product in chapter 2, but for now let's just take the basics.

The reason this modification was required was due to the x86 processor architecture and its concepts of security rings (ring 0 through 3). In virtualization the host operating system generally will run at ring level 0, the most secure part of the processor if you will, with Virtual Machines running at ring level 3. Of course operating systems (like Windows or Linux) are written so that they require full control of ring level 0. So when these OS's are run as a guest virtual machine they need to be modified so that the guest OS is not making ring level 0 calls (or in Windows system/kernel calls) to areas of the processor to which it doesn't have access. With the processor virtualization technology that has been introduced, it is basically a hardware change to allow for a "virtual Ring 0" if you will, labeled 'Ring 1.' The idea is that the manufacturer can make changes to move the required instructions up a level (to ring 1) and then the guest OS's are fooled into thinking they have full control of ring 0 all the time.

Before these processors were released, using Xen required that you modified the kernel in the guest OS and host. For Windows shops this was not an option and slowed the adoption of Xen as a virtualization solution. With Xen out,

most Windows environments adopted ESX server since it allows the VM to execute against the processors (direct execution) without modifications to the guest OS at all.

ESX did (and does in 3.0) what I like to think of as processor thunking. Remember when you ran 16 bits apps on a 32 bit system, and the process for remapping the 16 bit addresses to 32 bit addresses was termed as "thunking." Well I like to think of ESX doing the same thing, but on the processor side. ESX essentially allows the guest OS to see the processor as it is. It then allows direction execution on the processor (even to ring 0) from the guest OS. The trick here is that the ESX kernel also runs at ring level 0, and each time a call from the guest comes down, the kernel essentially has stopped executing itself on ring 0, allowing the execution of the command from the guest (there is a binary translation involved) then sending the results back to the guest and resuming its operations.

This may sound like a ton of overhead, but it's not. Sure there is some, but essentially both forms of virtualization perform about the same, and in either case there is still a need for some type of virtual machine monitor/hypervisor to manage the guests and their access to resources.

Which brings us to our next point, no matter what the hardware manufacturers do (processor, NIC, HBA or Memory), there will, for the foreseeable future, be a hypervisor like Xen or ESX. A software layer will be required to manage all these hardware devices and their unique one-use hypervisors. The decision you have to make is which piece of software to use. Xen and VMware ESX are the two "big name" players with Microsoft Virtual Server following behind until their Longhorn hypervisor is publicly available. We'll get into the difference between hosted and bare metal architectures in the following pages, but at least now you can put to rest the arguments from the guy two cubes over about how you should just wait for the hardware hypervisors to come out and explain the reality of virtualization to him. Since this book is an ESX book we will continue to focus on that from here out.

So what is "virtualization?"

Simply stated, VMware (the company) provides virtualization technology. All of their products (Workstation, ESX Server, and VMware Server) work in more-

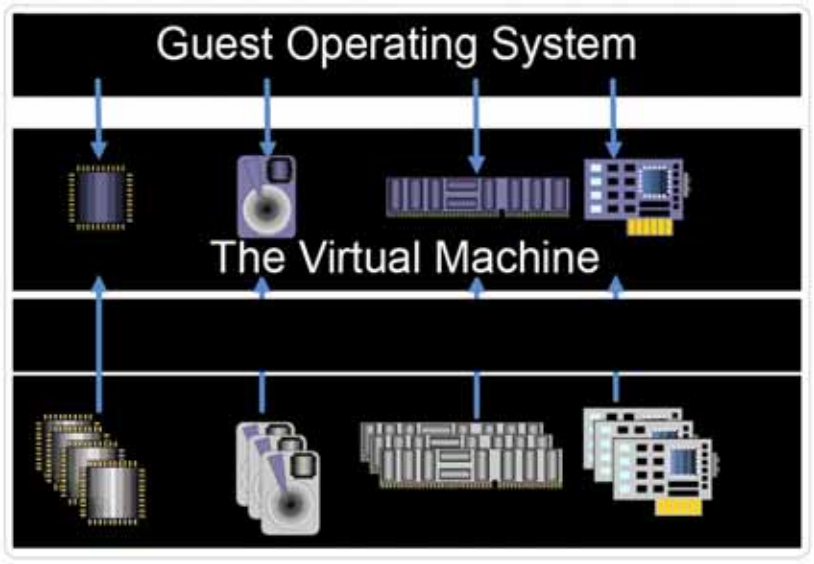
or-less the same way. There is a host computer that runs a layer of software that provides or creates virtual machines that x86 operating systems can be installed to. These virtual machines are really just that—complete virtual machines. Within a virtual machine you have hardware like processors, network cards, disks, COM ports, memory, etc. This hardware is presented through BIOS and is configurable just like physical hardware.

If we were to jump ahead a little and look at a virtual Windows server, you would be able to go into device manager and view the network cards, disks, memory, etc…, just like a physical machine. As a matter of fact the whole idea is that the virtualized operating system has no idea that it is on virtual hardware. It just sees hardware as it normally would.

In Chapter 2 we'll go into detail on how virtualization actually takes place with ESX Server, but for now it's just important to understand that from a high level there are a few components that make up any virtualization environment:

1. A host machine/host hardware
2. Virtualization software that provides and manages the virtual environment
3. The virtual machine(s) themselves (or "VM's"—virtual hardware presented to the guest
4. The guest operating system that is installed on the virtual machine

Figure 1.1: The basic virtualization model



Knowing that each of these four elements must be in place in a virtual environment and understanding that they are distinctly different resources will allow you to understand where different virtualization software is used. Let's examine each of these components a bit before moving into the breakdown of the different virtualization technologies available from VMware and other vendors.

The Host Machine

The host machine in a virtual environment provides the resources to which the virtual machines will eventually have access. Obviously, the more of the resources available the more virtual machines you can host. Or put more directly, "the bigger the host machine, the more virtual machines you can run on it." This really makes sense if you look at Figure 1.1. In Component 1 of the diagram, the host machine has processors, some RAM, a set of disks, and network cards. Assuming that the host is going to use some of each of these core four resources for its own operation, you have what's leftover available for the virtual machines you want to run on it.

For example, let's assume that the host is using 10% of the available processor for itself. That leaves 90% of the CPU available for the virtual machines. If you're only running a single virtual machine (VM) then this may be more than enough. However, if you're trying to run 30 VM's on that host then the hosts "extra" 90% CPU availability will probably lead to a bottleneck.

The other challenge is that since all of the VM's are sharing the same resource (the CPU's in this case), how do you keep them from stepping on each other? This is actually where Component 2 from Figure 1 comes in—the virtualization software.

The Virtualization Software

The virtualization software layer provides each virtual machine access to the host resources. It's also responsible for scheduling the physical resources among the various VM's. This virtualization software is the cornerstone of the entire virtualization environment. It creates the virtual machines for use, manages the resources provided to the VM's, schedules resource usage when there is contention for a specific resource, and provides a management and configuration interface for the VM's.

Again, we can't stress enough that this software is the backbone of the system. The more robust this virtualization software is the better it is at scheduling and sharing physical resources. This leads to more efficient virtual machines.

VMware provides three versions of this virtualization software. The first two—"VMware Workstation" and "VMware Server"—are virtualization software packages that install onto an existing operating system on a host computer. The third version ("VMware ESX Server") is a full operating system in-and-of itself. We'll explore some of the reasons to help you choose which product you should use later in this chapter. The important idea here is to understand that ESX Server is both its own operating system and also the virtualization software (Components 1 and 2 in the model we've been referencing), while VMware Server and Workstation are virtualization software packages that are installed on and rely upon other operating systems.

The Virtual Machine

The term "virtual machine" is often incorrectly used to describe both the virtual machine (Component 3) and the guest operating system (Component 4). For clarity in this book we will not mix the two. The virtual machine is actually the virtual hardware (or the combined virtual hardware and the virtual BIOS) presented to the guest operating systems. It's the software-based virtualization of

physical hardware. The guest operating systems that we install into these "machines" are (in most cases) unaware that the hardware they see is virtual. All that the guest OS knows is that it sees this type of processor, that type of network card, this much memory, etc.

It's important to understand that the virtual machine is not the OS but instead the hardware and configurations that are presented to the guest OS.

The Guest Operating System

In case it's not clear by now, the guest OS is the x86-based operating system (Windows, Linux, Novell, DOS, whatever) that's running on a VM. Again, understanding that the guest OS (or "guest machine" or simply "guest") is simply the software (Component 4) that's installed onto a VM (Component 3) will make your life easier when it comes to understanding and troubleshooting your environment.

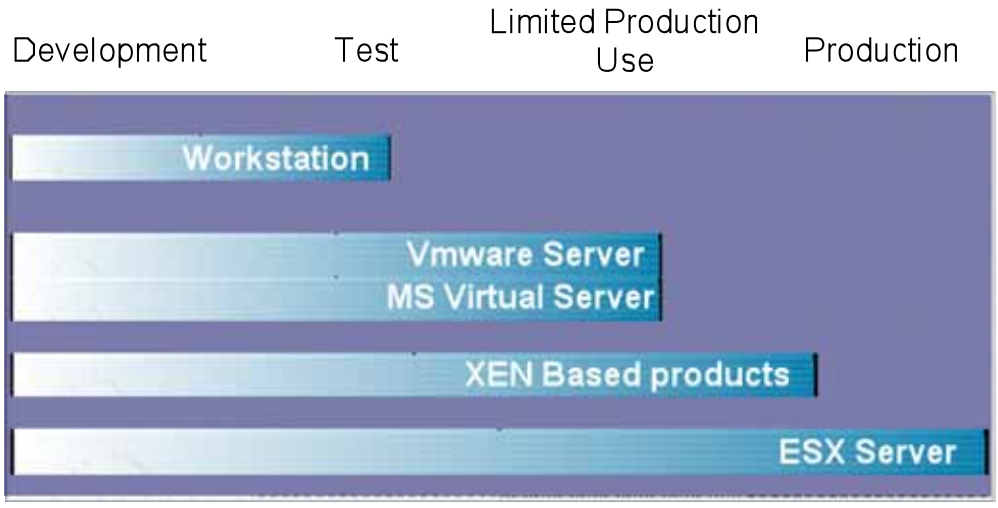
Once all four of these components are in place you'll have a virtual environment. How this virtual environment performs, is managed, and the types of functionality available in your virtual environment are all dependent on the type of software you're using to provide your virtual environment.

Which Virtualization product which should I use?

Well that's the question isn't it? Since this book is about ESX Server we can probably assume that you've already made your decision. Then again you might be using this book to help make your decision, so we'll go ahead and look at the products and how they fit into the IT world.

Most VMware folks started out using VMware Workstation (simply called "Workstation" by VMware geeks in the community). Workstation allows us to create virtual workstations and servers on our own PC. This lets us create small test environments that we can use to test scripts, new software packages, upgrades, etc. VMware Workstation is perfect for this.

Figure 1.2: Where do the VMware and other Virtualization products fit?



Of course Workstation has its limitations. Probably the biggest one is that VM's can only run while you're logged into your host workstation. Log off and the VM's shutdown. Also, VMware Workstation is pretty much a local user tool which means that there are really no remote administration capabilities whatsoever. These limitations keep Workstation on the desktops of developers, engineers, and traveling salespeople who have to give multi-server demos off their laptops. No one uses Workstation for production environments.

VMware Server (formerly called "GSX" for short) is a step up from Workstation. Server is basically a software package that installs onto an existing host operating system (either Linux or Windows Server). It offers remote management and remote console access to the VM's, and the various VM's can be configured to run as services without any console interaction required. Its limitation is really that it has to use resources from the host hardware through the host OS. This really limits the scalability and performance of Server.

The reason for this is that with Server, VM's do not have direct access to the hardware. Let's look at an example to see how this can cause a problem. We'll look at memory use. Let's assume you configure 384MB of memory for a VM running on VMware Server for Windows. The "catch" here is that since VMware Server is "just" another Windows application, the VM doesn't get direct access to 384MB of memory. Instead it requests 384MB of memory from Windows and is dependent on the Windows scheduling mechanism.

Sure you'll see the host's memory utilization go up by 384MB (plus some for overhead) when you turn on the VM, but the guest OS has to send all memory requests to Windows. In this case you'll have a host OS managing the "physical" memory for the guest OS. This is on top of the guest OS managing its own memory within the VM.

While this is just a simplified example, it points out some of the inherent limitations with VMware Server that aren't seen in ESX. Does this mean VMware Server isn't a good product? Not at all. It just means that it has limitations stemming from the fact the virtualization software runs on top of a host OS. VMware Server is still used in plenty of environments—especially those that don't require enterprise class scalability for their VM's, those that have a limited numbers of VM's, and those that do not require maximum performance. VMware Server is also frequently found in corporate test labs and is used to allow administrators to get the benefits of a "virtual" test environment without the them needing to know all the ins and outs of a full virtual server OS. Finally, many companies use VMware Server when they don't have the budget to buy ESX-certified hardware or when the VMware champions can't win the political "everything has to run on Windows" battle.

What makes ESX different than VMware Server, Workstation, or even Microsoft's Virtual Server in its current revision?

VMware ESX Server is its own operating system. Unlike VMware Server or Microsoft Virtual Server, ESX is not a software package that installs into a host OS- ESX is the host OS. Engineered from the beginning to be nothing more than a VM host, ESX Server is completely designed to give the VM's the best performance possible and to allow you (the admin) to control and shape the way the host resources are shared and utilized.

So what does using ESX instead of VMware Server or Workstation get you? The answer is simple: performance (more management and recovery features and reliability).

- Performance. ESX Server provides a level of performance for your VM's that simply cannot be found in VMware Server or Workstation. It also allows for more advanced resource allocation, fine tun-

ing of performance, a better VM-to-processor ratio, and more advanced resource sharing.

- Management and Recovery Features. ESX, when used with Virtual Center, allows you to manage the load on your systems by moving a VM to any host in a cluster without downtime. It also has an automated mode where it will move and shift load (by moving VM's) to different hosts in the cluster automatically. In the event of a server failure in the cluster, the VM's will be restarted on the remaining hosts and brought back online within minutes.
- Reliability. VMware published an ESX Server Hardware Compatibility List (HCL). If the hardware you're using for ESX is on the HCL, then you can be confident that everything will work as expected. ESX also lets you get rid of any problems that exist in the host OS since host OS's don't exist with ESX.

In short if you're looking to implement virtual machines on an enterprise level or if you're looking to host a lot of production servers as VM's, then ESX is a great choice.

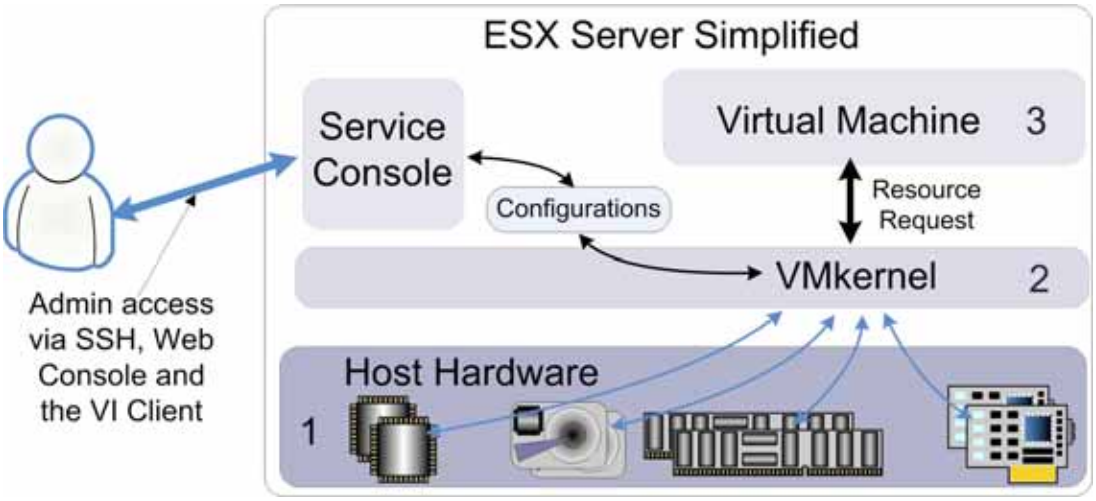
A 60-second Overview of the ESX Server Architecture

An ESX Server is made up of two core components:

- The ESX Server kernel (called "VMkernel")
- The Service Console

The term "ESX Server" is usually used to describe all of this stuff together.

Figure 1.3: ESX Server Simplified



There is quite a bit of confusion in regards to what the Service Console and the VMkernel really are. The service console is a customized Linux 2.4 kernel based on a Redhat Enterprise Linux distribution. The key to understanding the Service Console's relationship to the kernel is that the Console is really just a high priority virtual machine that allows you (though tools) to interact with the VMkernel and manage its configurations. The VMkernel on the other hand is the hypervisor and the real guts of ESX.

In Chapter 2 we'll go into great (and sometimes painful) detail about the VMkernel and the Service Console. For now we just want you to understand the basic architecture so you see how ESX is different from VMware's other two main products.

Referring to Figure 1.3 you can see that the service console is what allows us to interact with this server. This operating system allows us Secure Shell access, supports a web based management console, and allows us to manage the server. But, the service console is not ESX itself, it does not schedule resources or manage hardware access, and basically would be a simple Linux server if it wasn't for the VMkernel.

The VMkernel is what manages/schedules access to specific hardware resources on the host. It is the VMkernel that provides the Virtual Machines into which guest operating systems can be installed. This kernel is what makes ESX different from the other software packages available. The VMkernel allows direct

hardware access to the core 4 resources. It manages memory for the VM's, schedules processor time for the VM's, maintains virtual switches for VM network connectivity and schedules access to local and remote storage.

This kernel has been specifically built for this task. Unlike Windows or Linux hosts that have been built to be multi-purpose servers, this kernel's whole purpose is to share and manage access to resources. This makes it extremely light yet extremely powerful. Overhead in VMware ESX is estimated at 3-8%, while overhead for the host in these other OS's is generally 10-20% and sometimes as high as 30% depending on configurations.

The reduction in overhead basically comes from ESX being a "bare metal" product. Unlike the technologies used in workstation, Server or the current *Microsoft products, ESX makes the most of your hardware and has been built from the ground up to provide superb VM performance. Contrast this to the GSX, Workstation and Microsoft Virtual Server products that are really add-ons to operating systems that are built to handle numerous tasks and are not focused on providing high end VM performance.

*- a note here on the Microsoft virtualization product: At the time of writing Microsoft's publicly available virtualization product is MS Virtual Server 2005 R2. This product is an application that runs in a standard Windows 2003 Server environment. Microsoft's road map for virtualization includes a lightweight, bare-metal hypervisor like ESX Server, but at this point it is a road map available to the public and only beta code, so when comparing products we are using currently available technology.