ghaff@
illuminata.com

The Server Virtualization Bazaar, Circa 2007

Research Note

Gordon Haff
27 July 2007

Inspired by both industry hype and legitimate customer excitement, many companies seem to have taken to using the "virtualization" moniker more as the hip phrase of the moment than as something that's supposed to convey actual meaning. Think of it as "eCommerce" or "Internet-enabled" for the Noughts. The din is loud. It doesn't help matters that virtualization, in the broad sense of "remapping physical resources to more useful logical ones," spans a huge swath of technologies—including some that are so baked-in that most people don't even think of them as virtualization any longer.

However, one particular group of approaches is capturing an outsized share of the limelight today. That would, of course, be what's commonly referred to as "server virtualization." Although server virtualization is in the minds of many inextricably tied to the name of one company—VMware—there are many companies in this space. Their offerings include not only products that let multiple virtual machines (VMs) coexist on a single physical server, but also related approaches such as operating system (OS) virtualization or containers.

In the pages that follow, I offer a guide to today's server virtualization bazaar—which at first glance can perhaps seem just a dreadfully confusing jumble.

This Bazaar's Beginnings

About five years ago, I wrote a research note that considered the various approaches to splitting up physical servers so that they could run multiple applications in isolation from each other.[1] At that time, virtual machines were just one technique among many. They'd been around since the late 1960s on IBM mainframes, but had long played second-fiddle to more production-focused partitioning techniques that tilted towards isolation even at the expense of flexibility—a common tradeoff where workload management is concerned. However, even in 2002, it was clear that VMs were destined to become more important, given their introduction on volume Linux and Windows servers by VMware. By then, SWsoft had also released a product, Virtuozzo, which also let multiple applications run side-by-side on volume servers using a different technical approach—one we call "containers" or "OS virtualization."

Partitions enforced by hardware (physical partitions) and related techniques that are more software-based but still maintain a close correspondence between underlying hardware and partition boundaries remain important in scale-up system designs. But hardware-oriented styles of partitioning no longer dominate the discussion as they did a few years back.

In part, this is because smaller, higher-volume systems have become an ever-increasing segment of the computing landscape—even in roles that large RISC servers and mainframes once dominated. Physical partitioning, which typically can only carve systems into relatively coarse chunks, is of little interest in servers that are often smaller than the smallest physical partition would be. However, it's also the case that, although VMs were and are valued by users for their slicing and dicing prowess—which lets multiple applications be consolidated onto a single physical server—they are increasingly called on to do more. Because virtualization, almost by definition, provides some level of abstraction between applications and/or operating systems and what sits beneath, virtualization also makes it easier to then move those applications around. Under rubrics such as "virtual infrastructure," these techniques leverage virtualization to create an application environment that is far less tied to specific physical servers—and can thereby be more easily moved around in response to changes in usage patterns or problems with some underlying hardware.

Reducing costs by folding multiple underutilized servers into one (server consolidation) and being able to quickly fire up and tear down test environments remain important drivers of virtualization adoption. However, improved flexibility in deploying and reconfiguring applications is increasingly important as well.

With that as background, let's consider the two major approaches to volume server virtualization: VMs and operating system virtualization.

Virtual Machines

Virtual Machines (VMs) are software abstractions, a way to fool operating systems and their applications into thinking that they have access to a real (i.e. physical) server when, in fact, they have access to only a virtualized portion of one. Each VM then has its own independent OS and applications, and is not even aware of any other VMs that may be running on the same box, other than through the usual network interactions that systems have. Thus, the operating systems and applications from VMs are isolated from each other in the same manner as if they were running on separate physical servers. They're created by a virtual machine monitor (VMM), often called a "hypervisor," that sits on top of the hardware, where it creates and manages one or more VMs sitting on top of the hypervisor.

In "host-based" implementations, the VMM runs within a standard copy of an operating system. This has performance implications; any time the VMM needs to communicate with any I/O device, it needs to switch context back to the virtual machine application running atop the host OS, so that the host OS can do the actual I/O. This long I/O path

[1] See our *The Partitioning Bazaar: 2002*.

and the associated context switches carry with them a significant performance penalty for the I/O-intensive applications that characterize many heavy-duty server workloads. However, it has the important advantage of simplicity. The VM software is installed like any standard program. Furthermore, the standard operating system device drivers communicate with peripherals.

By contrast, a native hypervisor runs directly on top of the system hardware. It may be part of an operating system that has been modified to function as a hypervisor itself; this is the approach Microsoft will be taking, for example, when it builds virtualization into the next version of Windows Server. Alternatively, it may be a separate thinned-down standalone hypervisor that runs directly on top of the hardware in place of a conventional operating system. This is the case with VMware's ESX Server, for instance. Finally, the hypervisor may even be partly or wholly burned into firmware that ships with the system, as in the case of IBM System p LPARs.[2]

OS Virtualization, a.k.a. Containers

Containers[3] virtualize the system at the OS kernel layer. As with VMs, the applications running in each container believe that they have full, unshared access to their very own copy of that OS. Unlike VMs, containers exist with only one copy of the OS. There are no guest OSs.

Containers build additional separation onto the basic operating system process model. Although a process is not *truly* an independent environment, it does provide basic isolation and consistent interfaces. For example, each process has its own identity and security attributes, address space, copies of registers, and independent references to common system resources. These various features standardize communications between processes and help reduce the degree to which wayward processes and applications can affect the system as a whole. Over time, add-on resource management tools supported the ability to group processes into higher-level constructs such as "workloads" or "applications"—primarily to guarantee performance (or limit it) for different applications.

Containers build on this concept of workload groups by further isolating them from each other. A container may replicate a small subset of the operating system—mostly the libraries or writable data structures that can differ from one OS instance to another. However, for the most part, the containers running on a physical server (or within some form of partition,including a VM) share a single copy of the operating system.

Because containers run atop a single copy of the operating system, they consume very few system resources such as memory and CPU cycles compared to VMs, which require a full operating system image for each guest. They can simplify patches and other OS upgrades as well; again, there's only one OS instance so any changes can be immediately shared by all the application instances running on top. The downside? Well, a potentially big one is that there's only one OS instance. There's no way to run two different OSs on one server. Potentially more troublesome is that, while applications and libraries may vary from container to container, you can't even run multiple versions of the same OS on a single server to support applications that *need* different versions. These tradeoffs help to explain why hosting providers were early adopters of this virtualization approach; lower overhead translates to fewer servers and higher profits. At the same time, such organizations tend to run highly standardized environments without a lot of different versions of operating systems kicking around. Those with a greater need to change-manage divergent applications would be less well served.

---

2 In that firmware is just software burned into a chip, this isn't fundamentally different from a standalone hypervisor, although it may imply a smaller-footprint software layer that can fit into memory. Vendor-specific hypervisors also tend to extensively use the vendor's bespoke virtualization-assist instructions and embedded management processors.

3 The term "containers" is also sometimes used to refer to various types of application encapsulation, such as described in our *Trigence AE – Encapsulating Applications*. For our purposes here, I use containers to refer more narrowly to virtualization at the operating system level.

The x86 VM Landscape

With that as background, let's now take a look at some of the major players in virtual machine technologies for the x86 architecture (see Table 1).

The 800-pound gorilla is *VMware*, of course. VMware has evolved from a maker of tools for developers to a provider of a full range of infrastructure services for datacenters as well. Along the way, it was acquired by EMC but it remains—very atypically for an EMC property—largely independent. It will, in fact, soon be re-spinning out 10 percent of the company in an IPO.

VMware's flagship product is Virtual Infrastructure 3, a suite of products including Virtual Center and VMotion that build on the foundation of VMware's native hypervisor, ESX Server. The most intriguing news hasn't been announced, but has been spoken of by a number of sources. That's the idea of an "ESX light," an embedded hypervisor that will supposedly ship in at least some Dell servers and possibly those from others as well. Such a move would help to significantly increase virtualization's footprint in both the enterprise and SMB; it would also make it that much harder for vendors to capture revenue from base-level hypervisor products as VMs move one step closer to, if not commoditization (an overused and misused term), at least built-inness.[4]

The highest profile alternative to VMware is *Xen*. Xen itself isn't a product. It's an Open Source hypervisor project begun at the University of Cambridge. As originally conceived and implemented, it took an approach known as "para-virtualization"—meaning that it supports operating systems (OS) that have been modified to interact with the hardware through a virtualization layer. However, today, Xen implementations can take advantage of Intel's and AMD's virtualization assist hardware (known as Intel VT and AMD-V, respectively) to run unmodified (that is, not para-virtualized) guest operating systems.

Xen comes to market in two forms. The first is as technology integrated directly into the operating system. It now ships in both Novell and Red Hat Enterprise Linux products and will be making an appearance in OpenSolaris shortly. In this incarnation, one OS instance acts as the hypervisor and its management controller (called Dom0); additional OS instances can then run on top as guests. Xen is also sold as a standalone hypervisor by two companies: XenSource and Virtual Iron. XenSource is the commercial entity formed out of the Xen project. The company originally was focused on a grandiose policy-based automation effort, XenOptimizer, but has since scaled back efforts to focus more narrowly on providing a standalone hypervisor for Windows shops. Virtual Iron's current strategy is similar in many respects; it, too, refocused from initial efforts that included a proprietary hypervisor that could meld distributed systems into a single large SMP over InfiniBand.[5]

With Xen, the Open Source world seemingly coalesced around a single arrowhead, whereas previously it was unclear whether any of the fragmented, under-resourced Open Source server virtualization would ever achieve any sort of critical mass.[6] However, *KVM* (Kernel-based Virtual Machine) recently arrived on the scene, backed by stealth-mode startup Qumranet. Linus Torvalds has blessed the addition of KVM into the Linux kernel; Brian Stephens, Red Hat's CTO, has spoken warmly of KVM's approach of integrating directly into the Linux kernel as well. Xen, in contrast, is an independent component that nonetheless can require tight integration with the kernel—which can make keeping the two parts in sync difficult. KVM is still in early stages but could become of interest over the next two years depending upon how the server virtualization market evolves.[7]

[4] See our *Will We See an Embedded VMware "ESX Light"?*, *VMware—VMwhere?*, *EMC Looks to Have It Both Ways*, *VMware on the March*, and *VMware: Virtual Partitions for the Server Masses*.

[5] See our *Hypervisors in Boston*, *The Winding Road Toward x86 Virtualization*, *Virtual Iron's Early Steps*, and *Forging Irons for System-Spanning Virtualization*.

[6] E.g. User Mode Linux (UML), Plex86.

[7] See our *KVM Walks Onto the Field*.

Then there's *Microsoft*. Microsoft has been late, even shockingly so, to the server virtualization party. And it's been so determined to own the entire software pie that it's only grudgingly accommodated third-party server virtualization products—even though they bring with them plenty of opportunity to sell more Microsoft products. For now, Microsoft is making do with Virtual Server, a hosted VM product it picked up from Connectix. It's reasonably popular in Microsoft shops, but hardly represents the state-of-the-art in x86 virtualization. Microsoft's strategic play is to build virtualization into Windows Server 2008 ("Longhorn"). However, this technology, which goes by the codename "Viridian," won't be available until up to 180 days after the initial release of Longhorn; it has been jettisoning features to make even that tardy date. The only thing working in Microsoft's favor here is that, even if the company isn't thrilled to have people using other virtualization products, that isn't keeping its customers from doing just that—and buying Microsoft operating systems and applications to use with them.[8]

The VM Landscape (non-x86)

Virtual machines don't begin and end with the x86 architecture (see Table 2).

Indeed, what's probably today's most sophisticated implementation was also the first—IBM's z/VM on its System z mainframes. *z/VM*'s origins are intimately intertwined with early developments in time-shared computing that took place in Cambridge, Massachusetts during the early Sixties. z/VM's earliest roots are in CP-40, developed as a research project on a specially-modified IBM System/360 Model 40. CP-67 followed on a System/360 Model 67, IBM's first system with built-in virtual memory hardware. Although VMs on mainframes played second fiddle to IBM's more "production-oriented" operating systems for many years, they found new life as a path to efficiently supporting large numbers of simultaneous Linux instances on a single box. This has been a major factor behind IBM's mainframe resurgence.

Beyond the mainframe, most major system vendors offer some type of proprietary way to create virtual machines on at least parts of their product lines. IBM LPARs on its POWER-based servers were initially a software-based partitioning scheme that largely followed the outlines of underlying physical hardware such as CPUs and network cards. However, with the addition of capabilities such as sub-CPU "micropartitioning" and virtual networking—and the upcoming addition of Live Partition Mobility to its System p lineup in late 2007—it has,by any measure, evolved into a full-fledged virtualization approach.[9] HP covers the partitioning space with both physical partitions (nPars) and logical/virtual partitions (vPars); to this it has more recently added Integrity Virtual Machines. Integrity VMs have the fine granularity of vPars, but, like nPars, extend beyond the HP-UX version of Unix to also include current or planned support for Linux, Windows, and OpenVMS guests.

Sun's offering is Logical Domains (LDoms) on its UltraSPARC T1 "Niagara" servers. For now, this represents only a slice of Sun's SPARC-based product line, but that slice will expand as Sun rolls out more of its new generation SPARC processors —first, the upcoming "Niagara 2" and then the heftier "Rock" processor in 2008. For its part, Hitachi has "Virtage" for its Itanium-based BladeSymphony products. This product leverages Itanium's own flavor of hardware virtualization assist, VT-I, to both allow single blades to be divided up and multiple blades to be melded into a single large SMP server.[10]

In short, server virtualization is hardly just an x86 phenomenon. Just about every major datacenter server platform has some sort of virtualization play.

[8] See our *Microsoft Needs a Partner*.

[9] See our *It Slices, It Dices, It Runs a Lot Faster*, *Hitachi's Symphony of Blades*, and *Virtualization: Management Ascendant*.

[10] See our *Hitachi's Symphony of Blades*.

The Container Landscape

OS virtualization—*containers*—hasn't garnered the same level of attention that virtual machines have (see Table 3). However, they've proven popular for uses where minimizing overhead is a priority. Hosting providers are the canonical example of this use case, although we've also spoken with enterprise customers who have likewise gone the container route to minimize their use of hardware resources for virtualization.

SWsoft has clearly been the most prominent proponent of a standalone containers product. Founded in 1997, SWsoft released its Virtuozzo product in 2001. The company has historically very much concentrated on Web hosting companies where it has had considerable success, in part because it developed a rich set of complementary tools that cater to that market. However, more recently, SWsoft says that it has increasingly had success selling into the enterprise market as well, although that remains less than half its total business. The privately held company claims to be profitable and getting towards $100 million in annual revenues.[11]

The other most visible player in this technology space is Sun, whose Solaris Containers build namespace-isolation on top of the *zones* technology and administrative concept from Solaris resource management. (Indeed, Containers is essentially a Sun marketing name; technical documentation and the like all use the "zones" nomenclature instead.) For a time, in characteristic fashion, Sun seemed determined to pitch its containers as not only an alternative, but as a form of software-based partitioning *superior* to virtual machines for essentially all uses. Sun has since taken a more nuanced approach; containers have, to all appearances, proven popular within the Solaris base but Sun is now also planning to integrate Xen-based virtual machines into Solaris as a complementary virtual machine technology.[12]

Although Sun has led the charge towards containers among the remaining Unix vendors, both IBM and HP have or will have their own flavors. In HP's case, they're called Secure Resource Partitions and are based on HP's Process Resource Management (PRM) product with the addition of security isolation components from HP-UX Security Containment. For IBM's part, containers are part of the forthcoming AIX 6 in which they're called IBM Workload Partitions (WPAR). The neat twist with IBM's offer is the addition of container mobility—the ability to move a running container from one server to another—using technology that it acquired when it bought Meiosys.[13]

Conclusion

Server virtualization in its various forms has, within a remarkably short time, come from being an almost fringe technology to something that's often at the center of IT infrastructure discussions. One company, VMware, is no small part of the reason for that, and VMware will continue to be a very important player in this space for the foreseeable future.

But that doesn't mean than server virtualization begins and ends with VMware products. It doesn't even begin and end with the x86 architecture or with the specific virtual machine approach. It's a lot more complicated than that—which isn't a bad thing because the problems that server virtualization are being called on to solve aren't simple either.

[11] See our *Virtuozzo: The Lighter Side of Virtual Machines* and *New Containments for New Times*.

[12] See our *Sun Adds Another Virtualization Flavor* and *Solaris Rises*.

[13] See our *Stateful Transition: Coming to a Server Near You*.

Table 1. Virtual Machines for x86

Product/Technology	*Type*	*Host OS(s)*	*Needs HW Support?*	*Supported Guest(s)*	*Notes*
KVM	OS-based VMs	Linux	Y	Linux	Kernel-based alternative to Xen. Still under development.
Microsoft Virtual Server	Hosted VMs	Windows	N	Windows, Linux	The fruits of Microsoft's Connectix acquisition. Free.
Microsoft Windows Server virtualization "Viridian"	OS-based VMs	Windows Server 2008	Y	Windows, Linux	Scheduled for release within 180 days of Windows Server 2008. Some features have been cut.
Parallels Workstation (SWsoft subsidiary)	Hosted VMs	Mac OS X, Windows, Linux	N	Mac OS X, Windows, Linux, FreeBSD	More a consumer desktop play than VMware. Supposedly working on a server version.
Virtual Iron	Standalone hypervisor	None	Y	Windows, Linux	Initially built own hypervisor but switched to Xen in concert with own management products.
VMware ESX Server	Standalone hypervisor	None	N	Windows, Linux, Solaris, NetWare	Hypervisor foundation for VMware's Virtual Infrastructure 3 virtualization services.
VMware Server	Hosted VMs	Windows, Linux	N	Windows, Linux, Solaris	Free hosted product based on former VMware GSX Server.
VMware Workstation	Hosted VMs	Windows, Linux	N	Windows, Linux, Solaris, NetWare, FreeBSD	VMware's original product; developer-focused.
Xen Project	Integratable hypervisor technology	None	N (but req modified OS if not)	Varies by implementation	Currently the standard hypervisor technology used for Linux. Also used in standalone products from Virtual Iron and XenSource.
XenSource XenEnterprise	Standalone hypervisor	None	Y	Linux, Windows	Company was founded by the creators of the Xen project at the University of Cambridge.
XenSource XenServer	Standalone hypervisor	None	Y	Windows	Low-priced product version for Windows-only.

Table 2. Virtual Machines (non-x86)

Product/Technology	*Type*	*Host Platform(s)*	*Supported Guest(s)*	*Notes*
Hitachi Virtage	Hardware-based	BladeSymphony (Itanium)	Linux, Windows	Adapted from previous mainframe work. Expansion to Xeon blades planned.
HP Integrity Virtual Machines (IVM)	OS-based	HP-UX on Integrity (Itanium)	HP-UX, Windows, Linux	Part of HP Virtual Server Environment (VSE) together with nPars, vPars, and SRP partitioning.
IBM Logical Partitioning (LPAR) on System p	Hardware-based	IBM System p (POWER)	AIX, Linux	Originally more a partitioning technology à la System z LPARs. Evolved to be more "VM-like."
IBM z/VM	Hardware/OS-based	IBM System z	Linux, z/OS, VSE, CMS, z/VM	The ancestral VM. Roots go back to the mid Sixties in Cambridge, MA.
Sun Logical Domains (LDoms)	Hardware-based	SPARC (Sun Fire T1000/2000)	Solaris, Ubuntu Linux	These capabilities not available in APL line or UntraSPARC IV+ servers (many of which have physical partitioning as an alternative).
Xen Project	Integrate-able hypervisor technology	Depends on implementation	Varies by implementation	Although experimental code is available for a number of platforms, Xen is primarily an x86-centric technology.

Table 3. OS Containers

Product/Technology	*Host Platform(s)*	*Notes*
HP Secure Resource Partitions	HP-UX 11iv2 on Integrity	Based on HP's Process Resource Management (PRM) with security isolation provided through integration with the HP-UX Security Containment product.
IBM Workload Partitions (WPAR)	AIX 6 on System p	New with AIX 6. IBM has also largely overhauled their resource group management and implemented container mobility using technology from its Meiosys acquisition.
Linux VServer	Linux	An independent Open Source container project for Linux. Activity level fairly low.
Sun Solaris Containers	Solaris 10 (x86, SPARC)	Build on the "zones" in Solaris resource management. BrandZ (Branded Zones) is newer technology that allows Linux binaries to run on top of Solaris.
SWsoft Virtuozzo	Linux, Windows	Company largely got its start catering to hosting providers but has since started to diversify base. OpenVZ is complementary Open Source project.