

Xen, the virtual machine monitor

The art of virtualization

Moshe Bar

Virtualization is set to become a key requirement for every server in the data center. This trend is a direct consequence of an industry-wide focus on the need to reduce the Total Cost of Operation (TCO) of enterprise computing infrastructure. In spite of the widespread adoption of relatively cheap, industry standard x86-based servers, enterprises have seen costs and complexity escalate rapidly.

Virtualization is set to become a key requirement for every server in the data center

Today, for every dollar spent on computing hardware, as many as five dollars are spent on lifetime costs – support, maintenance, and software licenses. Operating System Virtualization, a concept pioneered by IBM in 1972 on the System 360, has become a key requirement, because it enables server consolidation, allowing multiple operating system and application images to share each server, cutting both hardware and lifetime costs.

But virtualization offers many, as yet, unrealized benefits – including development, staging and testing, dynamic provisioning, real-time migration, high availability and load balancing. Today's virtualization offerings are crippled by poor performance, lack of scalability, and an inability to offer the fine-grained resource guarantees that are required to provide true application level SLAs, and support dynamic load bal-

ancing and high availability. This article introduces Xen, a powerful, free software virtualization technology.

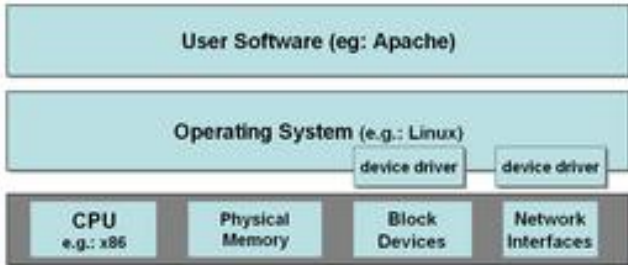
Virtualization: the new infrastructure requirement

The need for Operating System (OS) level virtualization has arisen as a result of a strange coincidence of market forces. First, enterprise software application architectures have become complex, multi-threaded, multi-process and multi-tiered systems, which are difficult to provision, configure and manage.

Second, the adoption of so-called “scale-out” computing infrastructure based on inexpensive, industry-standard servers, which has led to a proliferation of servers in the data center.

Frequently, IT staff provision one application per server, because it's the easiest way to ensure that the application and its configuration state can be isolated from other applications in the data center. Moreover, it provides a simple model for dealing with reliability and servicing – if the server fails, only the single application it hosts will fail. If the application must be protected against downtime during server maintenance, or from faults, then it's relatively straightforward to “clone” the entire state of a server, and copy it to an identical machine that can be brought into service to replace the system that goes offline. Finally, provisioning resources at the server level provides a way to identify the true resource needs of an application. If multiple

One App, One Box. On today’s servers, one operating system image, together with one application composed of multiple threads and processes, is tied to a single physical server. This leads to higher costs because each physical server requires maintenance and software licenses, and less flexibility because the application load is not matched to the server’s capacity, causing over/under utilization



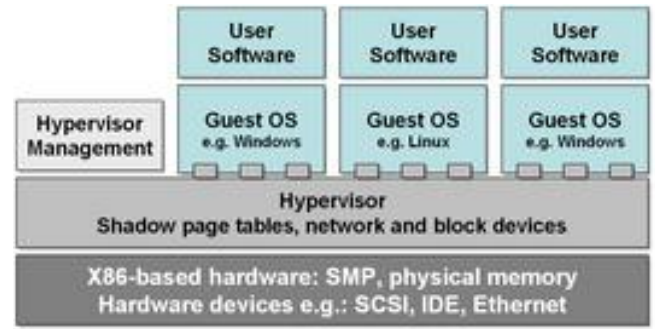
applications share a single server it’s difficult to determine the real resource needs of each, and to provision additional resources as needed.

Of course, serious drawbacks result from the apparent convenience of tying applications to the physical infrastructure. First, if the application demands less than the full capacity of the server, the CIO will quickly find that most servers are severely under-utilized (typically today, with the incredible capabilities of modern 2- or 4-way servers, utilization figures are about 10-15% per server - Gartner group, August 2004).

 Serious drawbacks result from the
 apparent convenience of tying
 applications to the physical infrastructure

Of course, each server consumes a full power load, and therefore requires cooling to match. But it also costs about five times as much to maintain – evenly split between the cost of software licenses and the cost of running the server. The net result: proliferation of under-utilized and expensive servers. Finally, the true benefits of scale-out computing are placed firmly out of reach: Easy maintenance, “dial-up/dial-down” provisioning of additional resources in response to the dynamically changing resource requirements of different applications, support for high availability and remote standby and handoff, and an ability to easily develop, test, stage and rapidly provision new applications

Emulated Virtualization. The guest OS is binary-rewritten to let the hypervisor intercept and manage all changes to hardware data structures, causing frequent address space context switches



across distributed data centers are all impossible without the help of OS virtualization.

What virtualization enables

OS virtualization is achieved by inserting a layer of software between the OS and the underlying server hardware. This layer is responsible for allowing multiple OS images (and their running applications) to share the resources of a single server. Each OS believes that it has the resources of the entire machine under its control, but beneath its feet, the virtualization layer transparently ensures that resources are properly shared between different OS images and their applications.

It is important not to confuse OS virtualization with so-called “application virtualization”, a software technique that in effect “bundles” all processes, threads and application related state for each different application hosted by an OS, into a virtual container. Application virtualization software vendors, such as Trigeance, attempt to provide balanced performance to each virtual container, by applying application-specific policies to the OS scheduler. This achieves few of the benefits of true OS virtualization, the least of which is its inability to take advantage of new hardware features for virtualization, and consequently is not considered a serious contender in the data center.

In OS virtualization, the virtualization layer (often called the hypervisor or Virtual Machine Monitor (VMM)) must manage all hardware structures, such as page tables, and I/O devices, DMA controllers and the like, to ensure that each OS, when running, sees a consistent underlying hardware

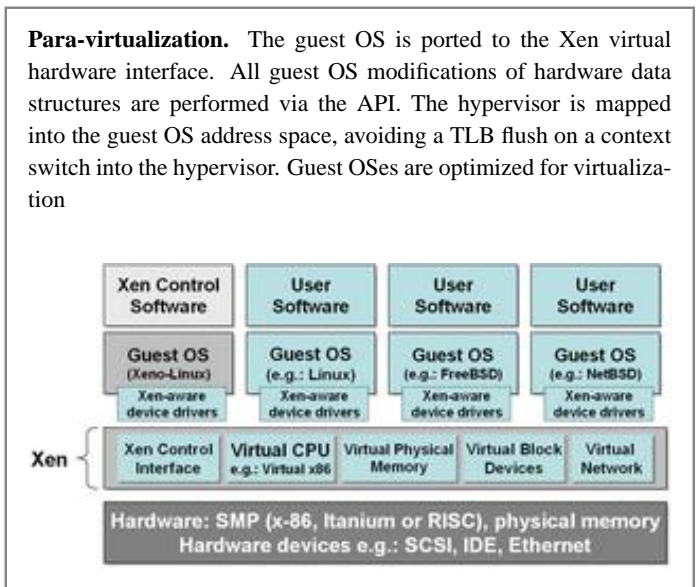
layer. Whenever the hypervisor performs a context switch between OS images, it must first preserve any state that the currently running OS will expect to be in place, in the hardware data structures, when its execution is later resumed, and then it must prepare the hardware for the next, incoming OS image. Of course, this comes at a price. The additional overhead that is required to manage all hardware states for the OS, and to present to it an idealized hardware abstraction causes a significant performance overhead. Because many hardware data structures, such as the Translation Lookaside Buffer (TLB), exist to speed up execution within the OS, when these are invalidated on a context switch, performance suffers dramatically because the incoming (newly running) OS image will fault on each page reference until the TLB is refreshed with its state.

It is important not to confuse OS virtualization with so-called “application virtualization”, a software technique that in effect “bundles” all processes, threads and application related state for each different application hosted by an OS, into a virtual container

There is another price too: vendors of virtualization software today charge a hefty premium (multiples of the server cost) for their software, to which must be added the usual OS and application costs. But while today’s virtualization products have allowed enterprises to realize significant benefits in the development, testing and QA of n-tier applications, a very high performance hypervisor is a requirement for production-grade server consolidation and to realize the promise of a more dynamic IT infrastructure. Xen, a free software hypervisor, is poised to deliver these benefits, because it outperforms existing hypervisors by an order of magnitude while providing guaranteed service levels to each guest OS. Furthermore, Xen is freely available as free software, and is being broadly supported by major industry players.

Xen: the best in virtualization, for free

Xen uses a very different technique than the hypervisors available today, namely para-virtualization. In para-



Para-virtualization. The guest OS is ported to the Xen virtual hardware interface. All guest OS modifications of hardware data structures are performed via the API. The hypervisor is mapped into the guest OS address space, avoiding a TLB flush on a context switch into the hypervisor. Guest OSes are optimized for virtualization

virtualization, the guest OS is ported to an idealized hardware layer, which completely virtualizes all hardware interfaces. When the OS updates hardware data structures, such as the page table, or initiates a DMA operation, it makes calls into an API that is offered by the hypervisor.

Xen fulfills the need for an unencumbered virtualization standard, and offers an opportunity to all players to take advantage of the massive trend towards dynamic datacenter management

This, in turn, allows the hypervisor to keep track of all changes made by the OS, and to optimally decide how to modify the hardware on any context switch. The hypervisor is mapped into the address space of each guest OS, minimizing the context switch time between any OS and the hypervisor. Finally, by co-operatively working with the guest OSes, the hypervisor gains additional insight into the intentions of the OS, and can make the OS aware of the fact that it has been virtualized. This can be a great advantage to the guest OS – for example the hypervisor can tell the guest that real time has passed between its last run, and its present run, permitting it to make smarter re-scheduling decisions to appropriately respond to a rapidly changing environment.

Para-virtualization provides significant benefits in terms of

device drivers and device interfaces. Essentially, device drivers can be virtualized using a para-virtualization model (by splitting the OS drivers into a “top” and “bottom” half), and running the bottom half as a separate domain, with memory, CPU and other resource guarantees. Moreover, the hypervisor itself is protected from bugs and crashes in device drivers, and can make use of any device drivers available on the market. Also, the virtualized OS image is much more portable across hardware, since the low levels of the driver and hardware management are modules that run under control of the hypervisor.

The net result is that Xen offers superb performance – typically more than an order of magnitude faster than any hypervisor on the market. The drawback of para-virtualization is that the guest OS must be ported to the idealized hardware interface. Of course, this is not an issue with operating systems such as Linux, Free BSD, and Solaris. But for closed source operating systems, a para-virtualized hypervisor must rely on hardware support for virtualization to ensure that the native binary of the guest OS can still share resources with other guest OSes.

Xen is a para-virtualizing hypervisor. It relies on one of two approaches to achieve fast virtualization:

1. Hypervisor replicated versions (in memory) of the above state, so that the guest OS is aware it doesn't have full access to and control of the CPU.
2. Hardware based CPU support for multiple guest OSes (replicated stack, task segment structure, GDT and flags) and (in future) support for I/O virtualization.

In the first approach, the hypervisor maintains in-memory copies of all hardware state, and transparently effects changes to the hardware data structures on a context switch, to ensure that the incoming guest OS sees consistent hardware state when it resumes operation. Careful management of state is required to ensure that the minimal set of changes is made to the hardware, to maximize efficiency. This is nowhere more important than in management of virtual memory, via the page table and TLB, by both the hypervisor and the ported OS.

In the second approach, the Xen hypervisor uses hardware based virtualization technologies such as Intel's Silverdale and Vanderpool Technologies (VT) or AMD's *Pacifica*. These new capabilities support multiple instances of

hardware state, one for each guest OS. Initial versions of this hardware provide CPU support for virtualization, but it is anticipated that in due course these capabilities will be extended into the chipset architectures, to support virtualization of I/O subsystems.

When the major chip vendors ship their CPU support for virtualization, Xen will be able to perform even better. Intel's VT, for example, introduces a software-managed TLB and Global Descriptor Table, which removes the need for Xen to replicate and control these structures, and for the OS to support virtualization of the page tables.

Virtualization and the promise of utility computing

For all the potential benefits of virtualization and utility computing, few enterprises have yet managed to achieve the levels of performance and support for a broad range of software and hardware that they desire. Xen fulfills that need.

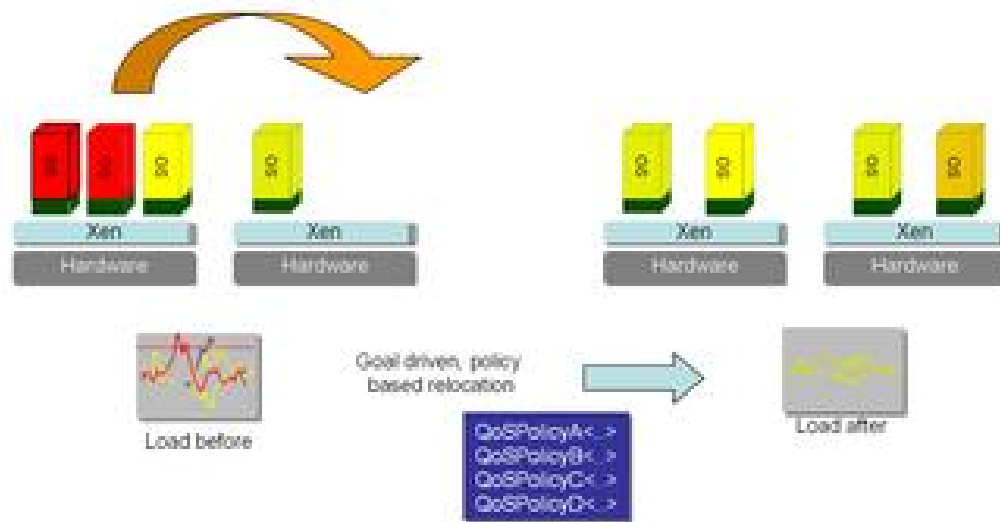
Xen uses a very different technique than the hypervisors available today, namely para-virtualization

With a high performance hypervisor, it will become possible to deliver on many of the key demands of major enterprises for an adaptive, responsive IT architecture. Xen supports a very wide range of hardware platforms, and therefore its guest OSes can run on a wide variety of hardware. Xen will soon support SMP guest operating systems, a key requirement for applications that today run on large SMP machines. Xen also offers a capability for live VM migration, in which a running guest OS, in its virtual machine, is moved to a second machine in a very short time.

While existing products on the market today claim live migration as a feature, they typically cause the migrated application to be unresponsive for tens of seconds while it is moved. Under Xen, with a feature that enables “copy on write” for guest OS pages, the “downtime” is typically 30-60ms, orders of magnitude faster than available today.

With these raw capabilities, Xen is ideally positioned to allow major enterprises to realize the promise of utility computing. Xen moves the level of infrastructure up above the basic hardware, by providing a common, low-level, high

Xen's Live Relocation. In a data center, Xen's live relocation capability can be used to move a running guest OS and application from one server to another, to achieve dynamic load balancing. This is done while the guest OS is running, with an almost imperceptible interruption in service for the moved image (about 30-60ms)



speed set of execution primitives that can be used to provide a dynamic and responsive computing environment.

The need for a free hypervisor

Today, several hypervisors are available on the market. None are free, and all are closed and tied into expensive, proprietary software stacks. Hardware vendors, rapidly moving to support virtualization, are naturally unhappy at the potential proliferation of virtualization technologies because it has the potential to slow down adoption. To fully take advantage of current and future virtualization features, the best technology should be widely adopted by the market. In addition, major enterprises want a virtualization layer that is not tied to any one OS, and that offers the best performance. Xen fulfills the need for an unencumbered virtualization standard, and offers an opportunity to all players to take advantage of the massive trend towards dynamic data-center management.

Xen is a free software project, run under the free software community rules. By virtue of its availability, and because it offers the best virtualization technology available, it is a natural candidate for a broadly adopted “standard” hypervisor. The free software community has embraced Xen as offering both the right technology – through its para-virtualization

approach and extremely high performance – and lack of bias towards any chip architecture, operating system or application vendor.

Copyright information

(The following license is effective immediately)

© by Moshe Bar

Verbatim copying and distribution of this entire article is permitted in any medium without royalty provided this notice is preserved.

About the author

Free software veteran and openMosix Project leader Moshe Bar is a founder and the CTO of XenSource, Inc. Prior to XenSource, Bar co-founded Qlusters, Inc., where he served as CTO, leading the company’s technology and product strategy. Previously, Moshe was VP, ERP implementations, at Baan Europe. He is the author of three books on Linux internals and free software development tools, a senior editor at byte.com, a founding research member of Democritos (the Italian national institute for nuclear simulation), and teaches at the UNESCO and U.N. Atomic Agencies.