



Xen — the Art of Virtualization A XenSource White Paper

Virtualization is set to become a key requirement for every server in the data center. This huge 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. 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 balancing and high availability. This white paper introduces Xen, a powerful, Open Source virtualization technology developed and maintained by the founders of XenSource, Inc., that has gained widespread support from across the industry. Xen is re-shaping the industry's approach to virtualization, catalyzing the adoption of so-called "utility computing".

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 that are difficult to provision, configure and manage. Second, the adoption of so-called "scale-out" computing infrastructure based on inexpensive industry-standard servers has led to a proliferation of servers in the data center. Frequently, IT staff provision one application per server, because that is 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, and if the application must be protected against downtime during server maintenance, or from faults, then it is 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 applications share a single server it is 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]). 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.

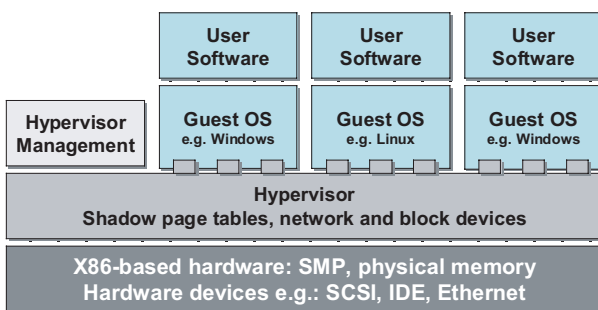
Easy maintenance, “dial-up/dial-down” provisioning of additional resources in response to the dynamically changing resource requirements of different applications, support for high availability and remote standby and handoff, and an ability to easily develop, test, stage and rapidly provision new applications 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 instances (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 container. Application virtualization attempts to provide balanced performance to the processes in each container by applying application-specific policies to the OS scheduler. This achieves few of the benefits of true OS virtualization, and consequently is not considered a serious contender in the data center.

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.

In OS virtualization, the virtualization layer (known as the *hypervisor*) 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 state 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. This so-called “emulated virtualization” relies on re-writing the existing operating system binary to ensure that the hypervisor can regain control of the system from time to time and whenever the OS modifies hardware data structures.

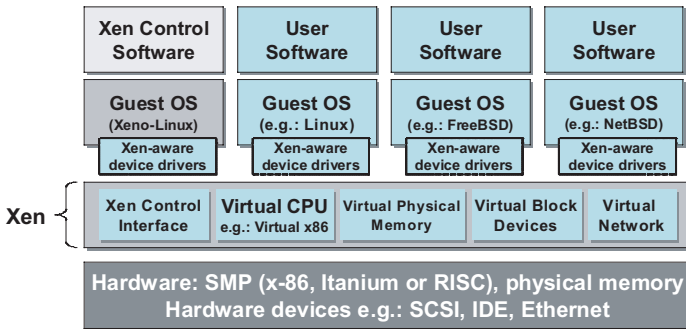
There is another price too: vendors of virtualization software today charge a hefty premium (multiples of the server cost) for their software, to which must be added the usual OS and application costs. But while today’s virtualization products have allowed enterprises to realize significant benefits in the development, testing and QA of n-tier applications, a very high performance hypervisor is a requirement for ubiquitous deployment of the hypervisor, and to realize the promise of a more dynamic IT infrastructure. Xen, the Open Source hypervisor developed and maintained by the founders of XenSource, Inc., is poised to deliver these benefits, because it outperforms existing hypervisors by up to an order of magnitude while providing guaranteed service levels to each guest OS. Moreover, Xen is freely available as Open Source 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-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. 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



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.

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. 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. Finally, the superior architecture of Xen ensures that the hypervisor itself is protected from bugs and crashes in device drivers, because drivers are run as separate protection domains from the core hypervisor. Borrowing from its Open Source lineage, Xen can make use of any device drivers available in existing software distributions.

The net result is that Xen offers superb performance - typically over 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 Open Source 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 does not 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.

The task is made easier if the CPU supports a software-managed TLB as these can be efficiently virtualized. A tagged or associative TLB is another useful feature supported by most server-class RISC architectures. Associating an address-space tag with each TLB entry allows the hypervisor and each guest OS to efficiently coexist in separate address spaces because there is no need to flush the entire TLB on a context switch. Unfortunately, the Intel x86 architecture does not have a software-managed TLB. The processor automatically services a miss by walking the page table in hardware. For best performance, all valid page translations should be present in the page table. Under Xen guest OSes are responsible for allocating and managing the hardware page tables, with minimal involvement from Xen to ensure safety and isolation. In addition Xen exists in a 64MB section at the top of every address space, thus avoiding a TLB flush when entering and leaving the hypervisor. Each time a guest OS inserts a new page table entry (for example on process creation), it registers it with Xen. At this point the OS must relinquish direct write privileges to the page-table memory: all subsequent updates must be validated by Xen. An OS may only map pages that it owns, and may not write to its page tables. Guest OSes may batch update requests to amortize the overhead of entering the hypervisor.

When the major chip vendors ship their CPU support for virtualization, expected in the second half of 2005, Xen will be able to perform even better. Intel's Silverdale and Vanderpool Technologies (VT) or AMD's *Pacifica*, will support multiple instances of hardware state, one for each guest OS. Intel's VT, for example introduces a software-managed Global Descriptor Table, which removes the

requirement for Xen to replicate this in software. It is anticipated that in due course the major chip vendors will extend their support for virtualization to include support for virtualization of I/O subsystems.

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.

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 migration of a running OS instance (also referred to as *Virtual Machine Migration*). This allows a running guest OS to be moved to a second machine in a very short time. While other 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 as low as 30-60ms, orders of magnitude faster than previously available.

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

The Need for an Open 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 datacenter management.

Xen is an Open Source project, run under the Open Source 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 Open Source 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.

XenSource, Inc.

XenSource, Inc., is a venture backed, Silicon Valley startup company founded by the architects of the Xen hypervisor, and backed by top tier investment firms Sevin Rosen Funds and Kleiner, Perkins, Caufield and Byers Partners (KPCB). The leader of the Xen project, Ian Pratt of the University of Cambridge, is a XenSource founder, and started the company specifically to promote the widespread adoption of Xen and answer its users’ needs for support, tapping enterprise software veteran Nick Gault as CEO, and Open Source veteran and openMosix leader Moshe Bar as CTO for their experience in serving a broad community base in an unbiased way. The Xen team at the University of Cambridge, will continue to maintain the source code tree, roadmap, and development of Xen.

XenSource has made a public commitment to work in an unbiased fashion with all industry partners to continually deliver the state of the art in hypervisor development to the Xen source code base, and to support all enterprises that seek to incorporate Xen in their product offerings. For more information visit www.XenSource.com or contact sales@XenSource.com.

XenSource, Inc.
500 Emerson Street
Palo Alto, California 94301
+1 650 326 0550 TEL
+1 650 326 0707 FAX