

The *Xen* of Virtualization

Assignment for CLC-MIRI

Amin Khan

Universitat Politècnica de Catalunya

March 4, 2013



Outline

- 1 Introduction
- 2 Architecture of Xen
- 3 Performance of Xen
- 4 Conclusion



Why Virtualization?

- Excess resources
 - ▶ Most tasks are limited by I/O not CPU
 - ▶ Similar to time sharing for multi-tasking
- Isolation
 - ▶ Security: Private memory for each process
 - ▶ Performance: A process shouldn't hang the whole system
- Heterogeneity
 - ▶ Multiple OS on a single hardware



Approaches to Virtualization

- Multiple Processes on standard OS
 - ▶ Performance Isolation is hard
 - ▶ System administration is difficult
- Full Virtualization
 - ▶ Provides complete abstraction of underlying hardware
 - ▶ Pros: No modification to host OS
 - ▶ Cons: Performance overhead
 - ▶ Limitation: Lack of hardware support in x86 architecture
 - ▶ Limitation: Difficult memory management



Paravirtualization

- Guest OS has *partially* direct access to physical hardware
- Better Performance (even without virtualization support in hardware)
- No need to emulate all hardware, so simpler to manage
- Pros: User applications can be executed without modifications
- Cons: Guest OS needs to be modified

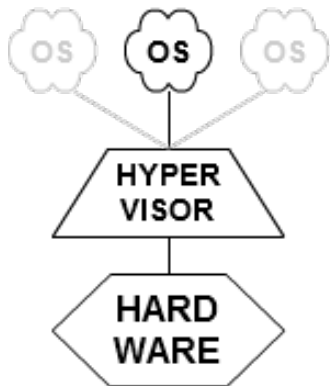


History

- Xen a breakthrough when no virtualization support in hardware
- Developed at University of Cambridge Computer Laboratory
- Supported by Microsoft. Windows XP's code was provided as well!
- Successful commercially, was acquired by Citrix Systems
- Commercial Products: Citrix XenServer, Oracle VM, Xen.org

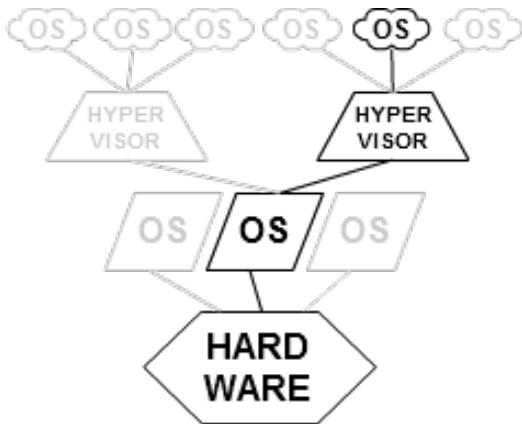


Xen: Bare-Metal Hypervisor



TYPE 1

native
(bare metal)

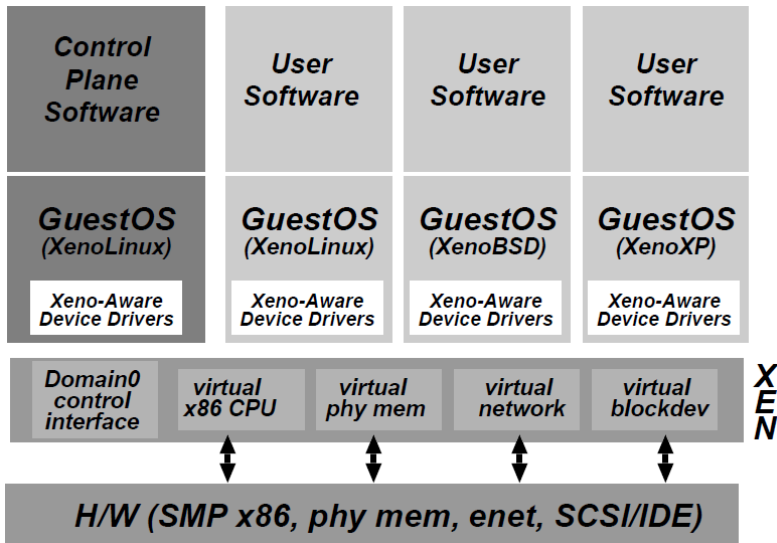


TYPE 2

hosted



Xen Architecture



Xen Virtual Machine Interface: Memory

- Virtualizing memory is hard!
- x86 doesn't have software-managed or tagged translation lookaside buffer (TLB)
- TLB maps virtual memory seen from a process address space to physical memory
- Xen gives control to guest OS to manage page tables for its processes



Xen Virtual Machine Interface: CPU

- Privilege levels, ring 0, 1, 2 and 3 in x86
- OS runs in separate address space and with higher privilege
- Xen hypervisor in ring 0, guest OS in ring 1, user processes in ring 3
- This requires modification to guest OS
- System calls efficiently handled as guest OS directly installs handler
- Page faults are slow as delivered through Xen



Xen Virtual Machine Interface: I/O

- Hardware devices are not emulated, just abstracted
- Data is transferred via Xen efficiently
- Uses shared-memory, asynchronous buffer-descriptor rings
- Light-weight event delivery mechanism to notify guest OS



Asynchronous I/O Handling

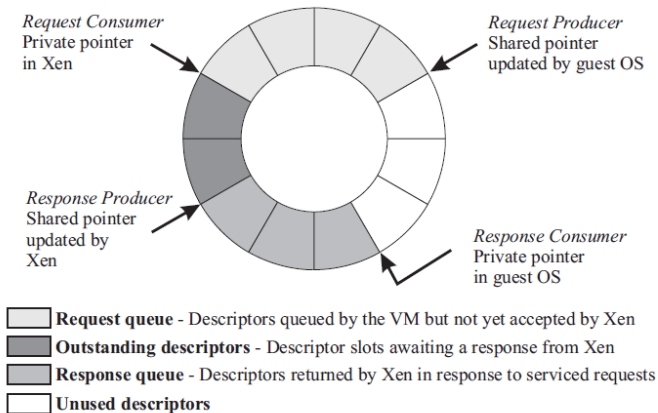


Figure 2: The structure of asynchronous I/O rings, which are used for data transfer between Xen and guest OSes.



Xen Virtual Machine Interface: Timers

- Real time: Nanoseconds since machine boot
- Virtual Time: Only advances when guest OS is executing
- Wall-clock Time: Off-set from current real time
- Guest OS can use pair of alarm timers, real and virtual



Performance Relative to Native Linux, VMWare, UML

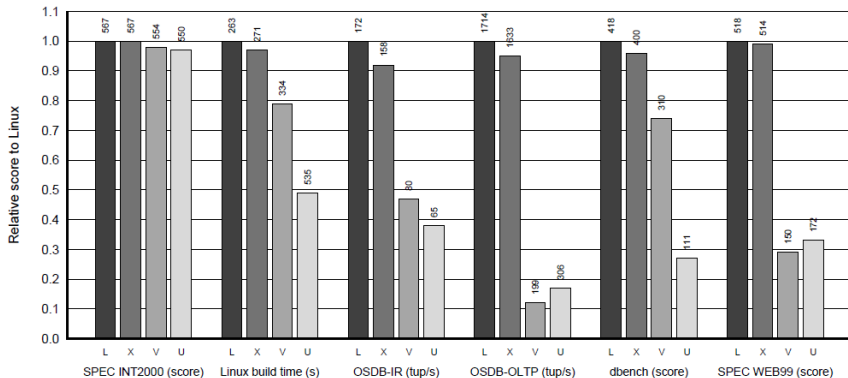


Figure 3: Relative performance of native Linux (L), XenLinux (X), VMware workstation 3.2 (V) and User-Mode Linux (U).



Operating Systems Benchmarks

Config	null call	null I/O	stat	openclose	slctTCP	sig inst	sig hndl	fork proc	exec proc	sh proc
L-SMP	0.53	0.81	2.10	3.51	23.2	0.83	2.94	143	601	4k2
L-UP	0.45	0.50	1.28	1.92	5.70	0.68	2.49	110	530	4k0
Xen	0.46	0.50	1.22	1.88	5.69	0.69	1.75	198	768	4k8
VMW	0.73	0.83	1.88	2.99	11.1	1.02	4.63	874	2k3	10k
UML	24.7	25.1	36.1	62.8	39.9	26.0	46.0	21k	33k	58k

Table 3: **lmbench**: Processes - times in μ s

Config	2p 0K	2p 16K	2p 64K	8p 16K	8p 64K	16p 16K	16p 64K
L-SMP	1.69	1.88	2.03	2.36	26.8	4.79	38.4
L-UP	0.77	0.91	1.06	1.03	24.3	3.61	37.6
Xen	1.97	2.22	2.67	3.07	28.7	7.08	39.4
VMW	18.1	17.6	21.3	22.4	51.6	41.7	72.2
UML	15.5	14.6	14.4	16.3	36.8	23.6	52.0

Table 4: **lmbench**: Context switching times in μ s

Config	0K File		10K File		Mmap Prot	Page
	create	delete	create	delete	lat	fault
L-SMP	44.9	24.2	123	45.2	99.0	1.33
L-UP	32.1	6.08	66.0	12.5	68.0	1.06
Xen	32.5	5.86	68.2	13.6	139	2.73
VMW	35.3	9.3	85.6	21.4	620	7.53
UML	130	65.7	250	113	1k4	21.8

Table 5: **lmbench**: File & VM system latencies in μ s



Network Performance

	TCP MTU 1500		TCP MTU 500	
	TX	RX	TX	RX
Linux	897	897	602	544
Xen	897 (-0%)	897 (-0%)	516 (-14%)	467 (-14%)
VMW	291 (-68%)	615 (-31%)	101 (-83%)	137 (-75%)
UML	165 (-82%)	203 (-77%)	61.1(-90%)	91.4(-83%)

Table 6: `ttcp`: Bandwidth in Mb/s



Concurrent Virtual Machines

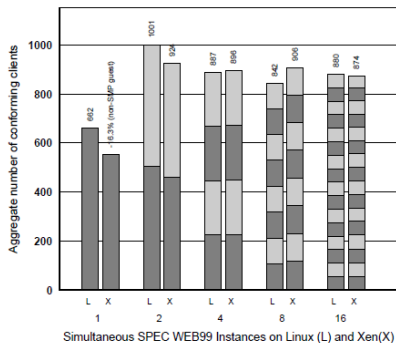


Figure 4: SPEC WEB99 for 1, 2, 4, 8 and 16 concurrent Apache servers: higher values are better.

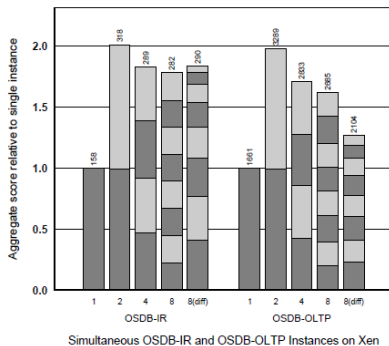


Figure 5: Performance of multiple instances of PostgreSQL running OSDB in separate Xen domains. 8(diff) bars show performance variation with different scheduler weights.



Scalability

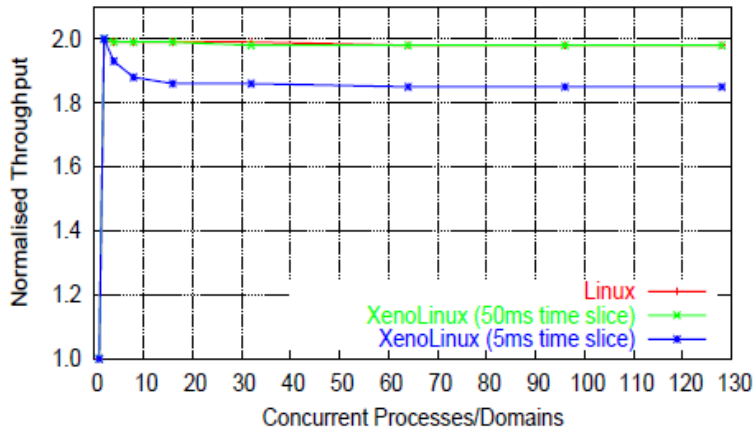


Figure 6: Normalized aggregate performance of a subset of SPEC CINT2000 running concurrently on 1-128 domains



Conclusion

- Supports Paravirtualization with modified guest OS
- Hardware-assisted virtualization, allowing for unmodified guest OS
- Virtual machine migration
- Available as dedicated virtualization platform
- Available as optional configuration for Unix, Linux, and Solaris systems
- QubeOS uses Xen for desktops to provide *Security by Isolation*

