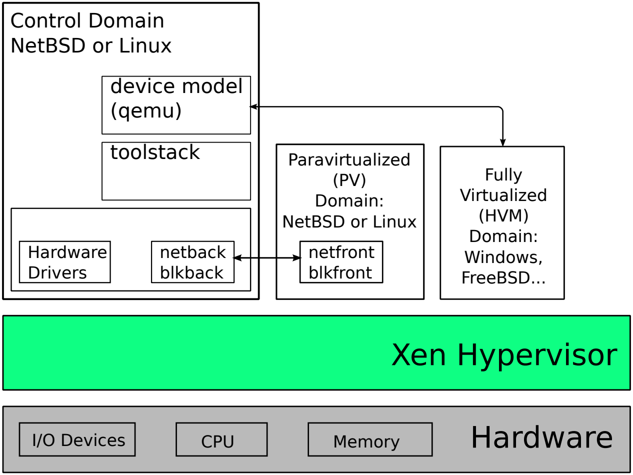
Benefits of the new Xen paravirtualization mode

Roger Pau Monné

Brussels – February 2-3, 2013

Xen.org

Xen Architecture



Paravirtualization

- Virtualization technique developed in the late 90s
- Designed by:
  - XenoServer research project at Cambridge University
  - Intel
  - Microsoft labs
- x86 instructions behave differently in kernel or user mode, options for virtualization were full software emulation or binary translation.
  - Design a new interface for virtualization
  - Allow guests to collaborate in virtualization
  - Provide new interfaces for virtualized guests that allow to reduce the overhead of virtualization
- The result of this work is what we know today as paravirtualiztion

Paravirtualization

- All this changes lead to the following interfaces being paravirtualized:
  - Disk and network interfaces
  - Interrupts and timers
  - Boot directly in the mode the kernel wishes to run (32 or 64bits)
  - Page tables
  - Privileged instructions

Paravirtualization

- The paravirtualization model works well on 32bit systems, but it has problems on 64bits
  - AMD decided to remove segmentation limit in 64bits, that was used by Xen to protect hypervisor memory
  - The 64bit architecture only has two memory protection levels
  - This is bad for the Xen architecture, three protection levels are needed in order to provide isolation between the hypervisor, the guest kernel and the guest userspace.
  - In 64bit PV guests every system call bounces up to Xen that performs the context switch to the guest kernel.

Paravirtualization

- Pros of paravirtualization
  - Provides near to bare metal speed
  - Reduces overhead of virtualization
- Cons of paravirtualization
  - Requires heavy modifications to the guest OS

Full virtualization

- With the introduction of hardware virtualization extensions Xen is able to run unmodified guests
- This requires emulated devices, which are handled by Qemu
- Makes use of nested page tables when available.

Full virtualization

- Pros of full virtualization
  - Doesn't require guest OS changes
- Cons of full virtualization
  - Slow IO because of emulated devices
  - Legacy boot
  - Need to run Qemu for each guest

The full virtualization spectrum

VS	Software virtualization
VH	Hardware virtualization
PV	Paravirtualized

🟥 Poor performance

🟨 Room for improvement

🟩 Optimal performance

	Disk and network	Interrupts and timers	Emulated motherboard	Privileged instructions and page tables
HVM	🟥 VS	🟥 VS	🟨 VS	🟩 VH
HVM with PV drivers	🟩 PV	🟥 VS	🟨 VS	🟩 VH
PVHVM	🟩 PV	🟩 PV	🟨 VS	🟩 VH
PV	🟩 PV	🟩 PV	🟩 PV	🟨 PV

Guest support in BSD

- List of BSD systems and operation modes:

	PV	PVHVM	HVM with PV drivers	HVM
NetBSD	YES	NO	NO	YES
FreeBSD	NO	YES	YES	YES
OpenBSD	NO	NO	NO	YES
DragonflyBSD	NO	NO	NO	YES

- Main problems:
  - Only PV guests are allowed as control domains (Dom0)
  - Some providers will only allow PV guests, or charge extra for HVM guests

Evolving Xen PV

- When PV was designed there were no virtualization extensions in hardware
- Everything is done using PV interfaces
- Hardware virtualization can provide speed improvements and ease of implementation
- Previous slides show how HVM mode evolved to use PV components
- But what happens if PV evolves to use HVM components?

Evolving Xen PV

VS	Software virtualization
VH	Hardware virtualization
PV	Paravirtualized

🟥	Poor performance
🟨	Room for improvement
🟩	Optimal performance

	Disk and network	Interrupts and timers	Emulated motherboard	Privileged instructions and page tables
HVM	🟥 VS	🟥 VS	🟨 VS	🟩 VH
HVM with PV drivers	🟩 PV	🟥 VS	🟨 VS	🟩 VH
PVHVM	🟩 PV	🟩 PV	🟨 VS	🟩 VH
PVH	🟩 PV	🟩 PV	🟩 PV	🟩 VH
PV	🟩 PV	🟩 PV	🟩 PV	🟨 PV

PVH

- Major features:
    - Runs in Ring 0
    - Uses PV entry point, skips BIOS emulation
    - Uses PV event channels, no APIC emulation
    - Native Page tables
    - Native IDT
- Development done by Mukesh Rathor at Oracle
- Main focus is increasing performance in 64bit PV guests

PVH

- Benefits for the BSD world:
  - Uses PVHVM callbacks for events
  - Doesn't use the PV MMU
  - Can be used as control domain (Dom0)
- The main benefit for BSD systems is that PVH is going to simplify one of the most difficult aspects of PV, the MMU
- Since the MMU is virtualized by hardware, porting new OSes to run in the PVH mode is greatly simplified

Conclusions

- Xen offers different guest virtualization modes
- Until now, PV was the only allowed mode for control domains
- PVH can be used as control domain, and simplifies the porting efforts
- The MMU changes, that where OS dependant are no longer necessary
- Drivers code can be shared between different BSD systems
- Transition from PVHVM to PVH is simpler than PV

Q&A

Thanks

Questions?