Xen and the Art of Virtualization *Revisited*

Ian Pratt, Citrix Systems Inc.

Outline

- A brief history of Xen
- Why virtualization matters
- Paravirtualization review
- Hardware-software co-design
  - MMU virtualization
  - Network interface virtualization

The Xen Story

- Mar 1999 XenoServers HotOS paper
- Apr 2002 Xen hypervisor development starts
- Oct 2003 Xen SOSP paper
- Apr 2004 Xen 1.0 released
- Jun 2004 First Xen developer’s summit
- Nov 2004 Xen 2.0 released
- 2004 Hardware vendors start taking Xen seriously
- 2005 RedHat, Novell, Sun and others adopt Xen
- 2006 VMware and Microsoft adopt paravirtualization
- Sep 2006 First XenEnterprise released
- May 2008 Xen embedded in Flash on HP/Dell servers

Xen Project Mission

- Build the industry standard open source hypervisor
  - Core "engine" that is incorporated into multiple vendors' products
- Maintain Xen's industry-leading performance
  - Be first to exploit new hardware acceleration features
  - Help OS vendors paravirtualize their OSes
- Maintain Xen's reputation for stability and quality
  - Security must now be paramount
- Support multiple CPU types; big and small systems
  - From server to client to mobile phone
- Foster innovation
- Drive interoperability

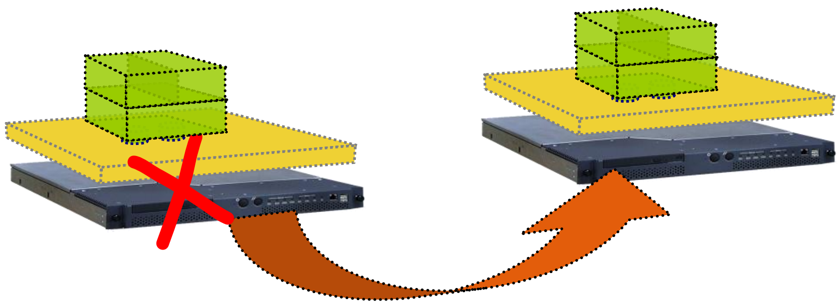
Why Virtualization is ‘Hot’

- Clearing up the mess created by the success of ‘scale-out’
  - One Application per commodity x86 server
  - Leads to ‘server sprawl’
  - 5-15% CPU utilization typical
- Failure of popular OSes to provide
  - Full configuration isolation
  - Temporal isolation for performance predictability
  - Strong spatial isolation for security and reliability
  - True backward app compatibility

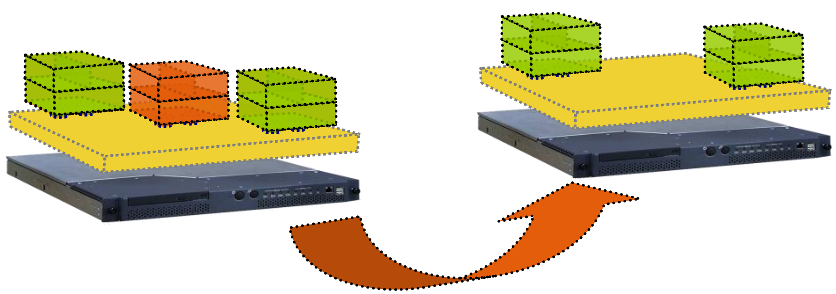
First Virtualization Benefits

- Server consolidation
  - Consolidate scale-out success
  - Exploit multi-core CPUs
- Manageability
  - Secure remote console
  - Reboot / power control
  - Performance monitoring
- Ease of deployment
  - Rapid provisioning
- VM image portability
  - Move image between different hardware
  - Disaster Recovery

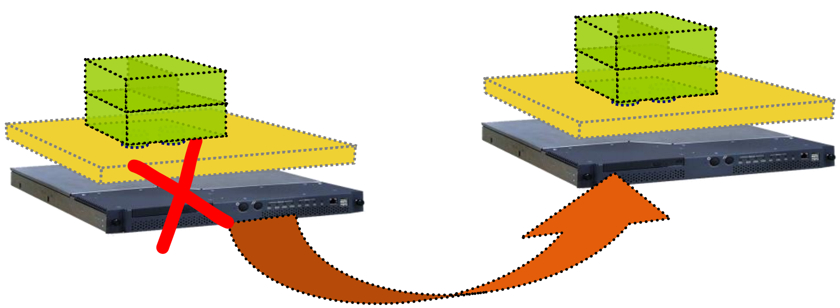
2nd Generation Virtualization Benefits



- ***Avoid planned downtime*** **with VM Relocation**



- **Dynamically *re-balance workload* to meet application SLAs**



- ***Hardware Fault Tolerance*** **with replay / checkpointing**

Hypervisor Security

- “hidden hypervisor” attack is a myth, but exploitation of an installed hypervisor is a real and dangerous threat
- Hypervisors add more software and thus increase the attack surface
  - Network-facing control stack
  - VM containment
- Hopefully much smaller and defensible than a conventional OS
  - Need a “strength in depth” approach
  - Measured launch

Improving Security with Hypervisors

- Hypervisors allow administrative policy enforcement from outside of the OS
  - Firewalls, IDS, malware scanning etc
    - More robust as not so easily disabled
    - Provides protection within a network rather than just at borders
  - Hardening OSes with immutable memory, taint tracking, logging and replay
  - Backup policy, multi-path IO, HA, FT etc
    - Availability and Reliability
- Reducing human effort required to admin all the VMs is the next frontier

Breaking the bond between OS and h/w

- Simplifies Application-stack certification
  - Certify App-on-OS; OS-on-HV; HV-on-h/w
  - Enables Virtual Appliances
- Virtual hardware greatly reduces the effort to modify/create new OSes
  - Application-specific OSes
    - Slimming down and optimization of existing OSes
    - “Native execution” of Apps
- Hypervisors enable h/w vendors to ‘light up’ new features more rapidly

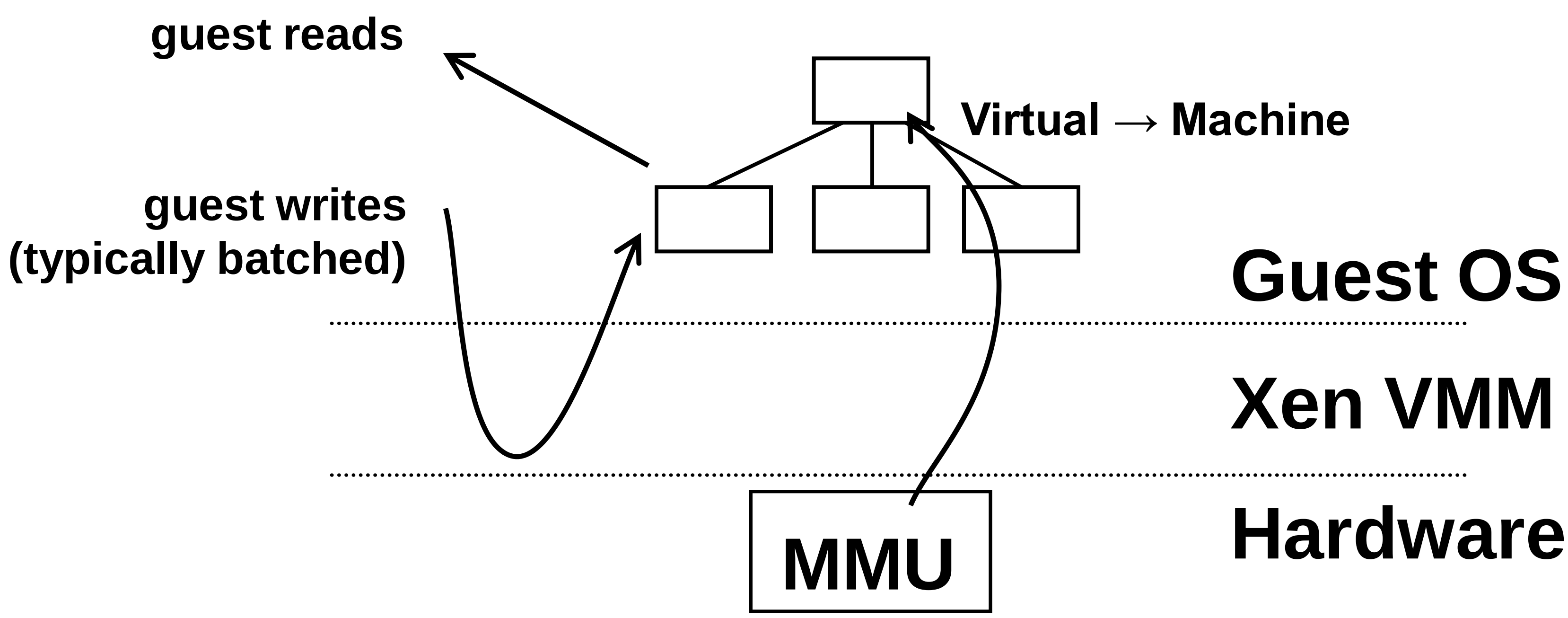
Paravirtualization

- Extending the OS to be aware it is running in a virtualized environment
  - For performance and enhanced correctness
  - IO, memory size, CPU, MMU, time
- In Xen <2.0, some paravirtulizations were compulsory to close x86 virtualization holes
  - Intel VT / AMD-V allow incremental paravirtualization
- Paravirtualization is still very important for performance, and works along side enhancements to the hardware
  - Higher-level paravirtualizations yield greatest benefit

MMU Virtualization

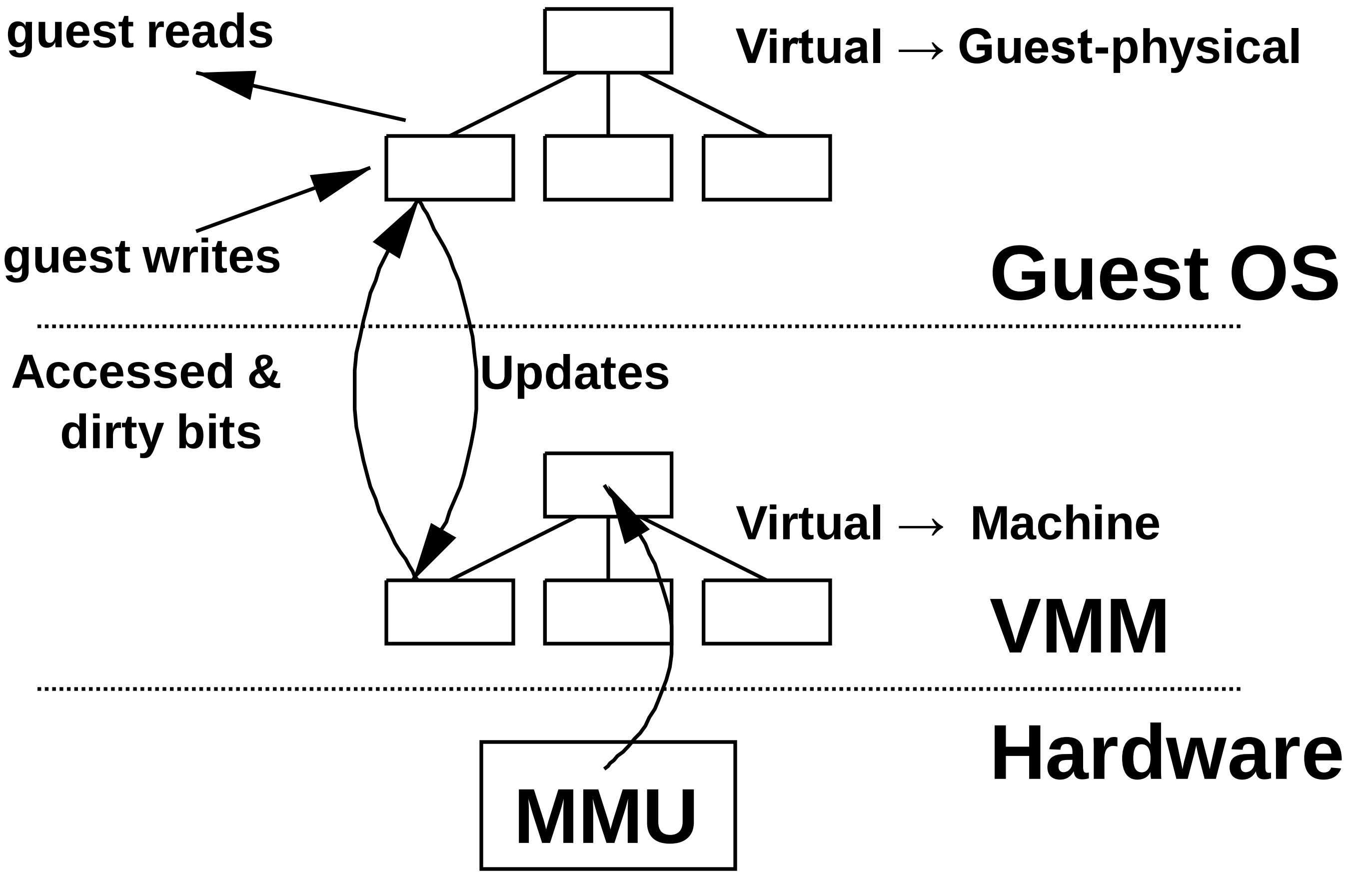
- Critical for performance, challenging to make fast, especially SMP
  - Hot-unplug unnecessary virtual CPUs
  - Use multicast TLB flush paravirtualizations etc
- Xen supports 3 MMU virtualization modes
  1. Direct pagetables
  2. Shadow pagetables
  3. Hardware Assisted Paging
- OS Paravirtualization compulsory for #1, optional (and very beneficial) for #2&3

MMU Virtualization : Direct-Mode



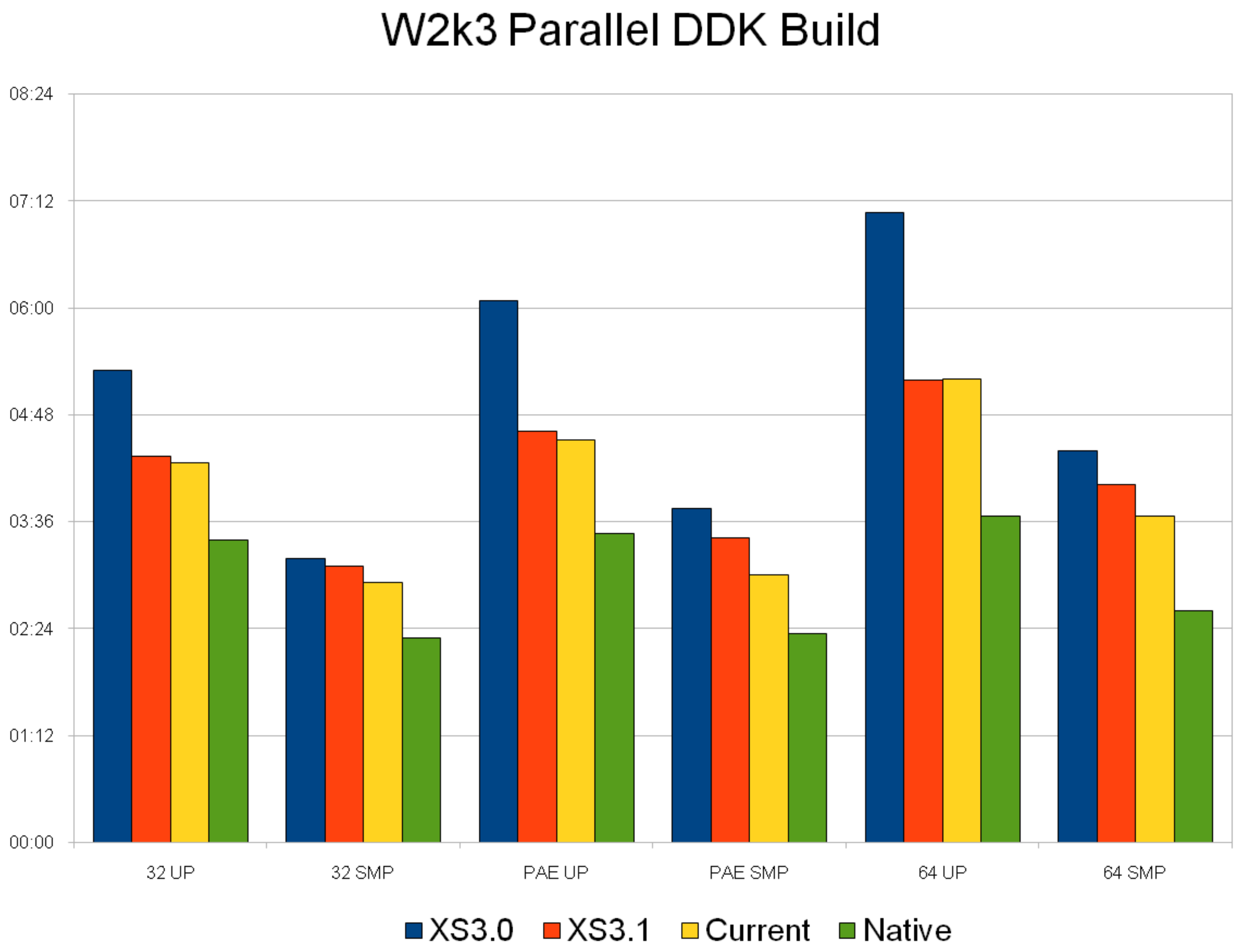
- Requires guest changes
  - Supported by Linux, Solaris, FreeBSD, NetBSD etc
- Highest performance, fewest traps

Shadow Pagetables



- Guest changes optional, but help with batching, knowing when to unshadow
- Latest algorithms work remarkably well

Peformance


W2k3 Parallel DDK Build
08:24
07:12
06:00
04:48
03:36
02:24
01:12
00:00
32 UP
32 SMP
PAE UP
PAE SMP
64 UP
64 SMP
XS3.0
XS3.1
Current
Native

Hardware Assisted Paging

- AMD NPT / Intel EPT
- Hardware handles translation with nested pagetables
  - guest PTs managed by guest in normal way
  - guest-physical to machine-physical tables managed by Xen
- Can increases the number of memory accesses to perform a TLB fill pagetable walk by factor of 5 (gulp!)
  - Hopefully less through caching partial walks
  - But reduces the effective TLB size
- Current implementations seem to do rather worse than shadow PTs (e.g. 15%)
  - Wide-SMP guests do relatively better due to no s/w locking
    - TLB flush paravirtualizations essential
  - H/w will improve: TLBs will get bigger, caching more elaborate, prefetch more aggressive

Network Interface Virtualization

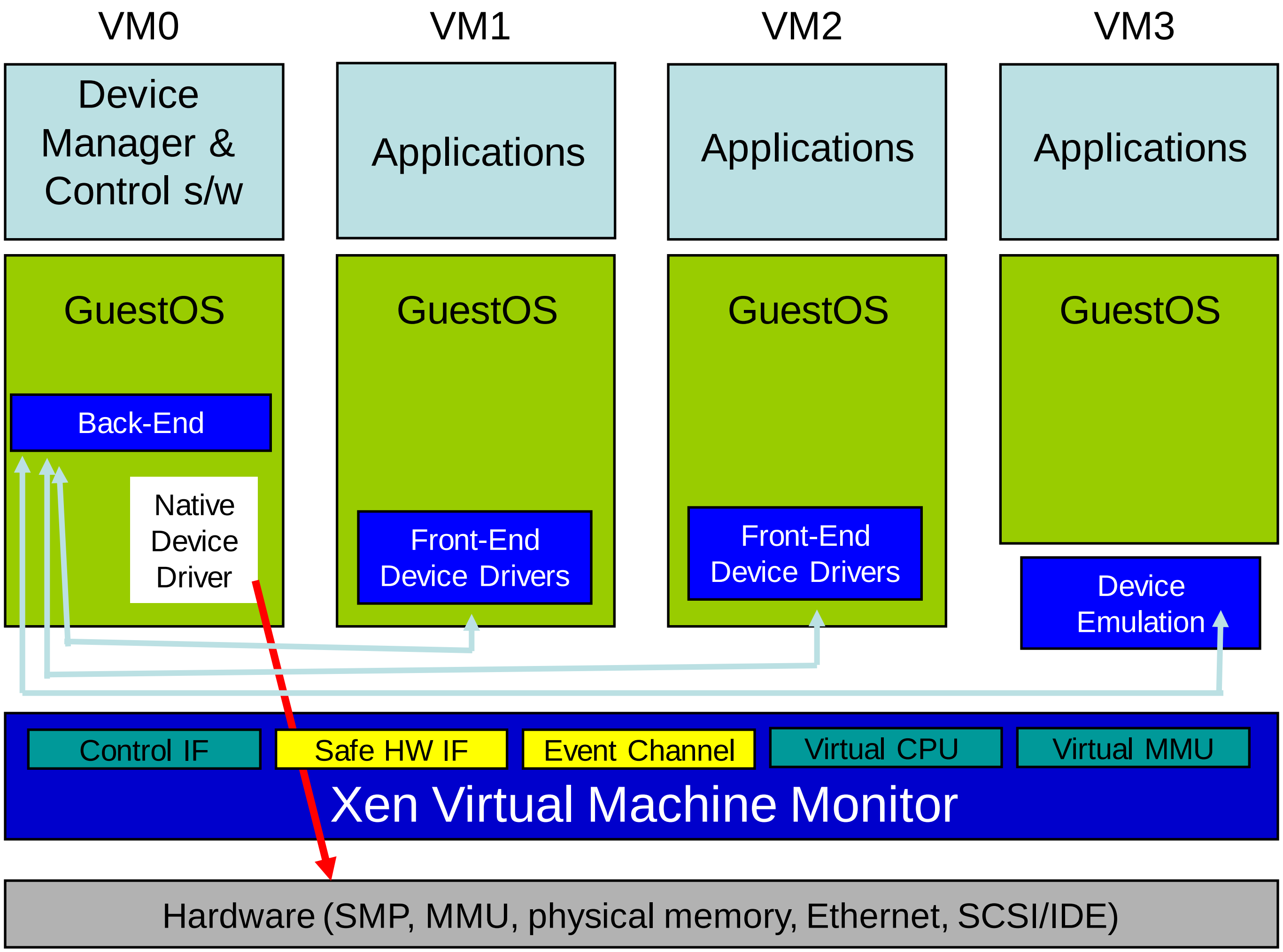
- Network IO is tough
  - High packet rate
    - Batches often small
  - Data must typically be copied to VM on RX
  - Some apps latency sensitive
- Xen's network IO virtualization has evolved over time
  - Take advantage of new NIC features
  - Smart NIC categorization: Types 0-3

Level 0 : Modern conventional NICs

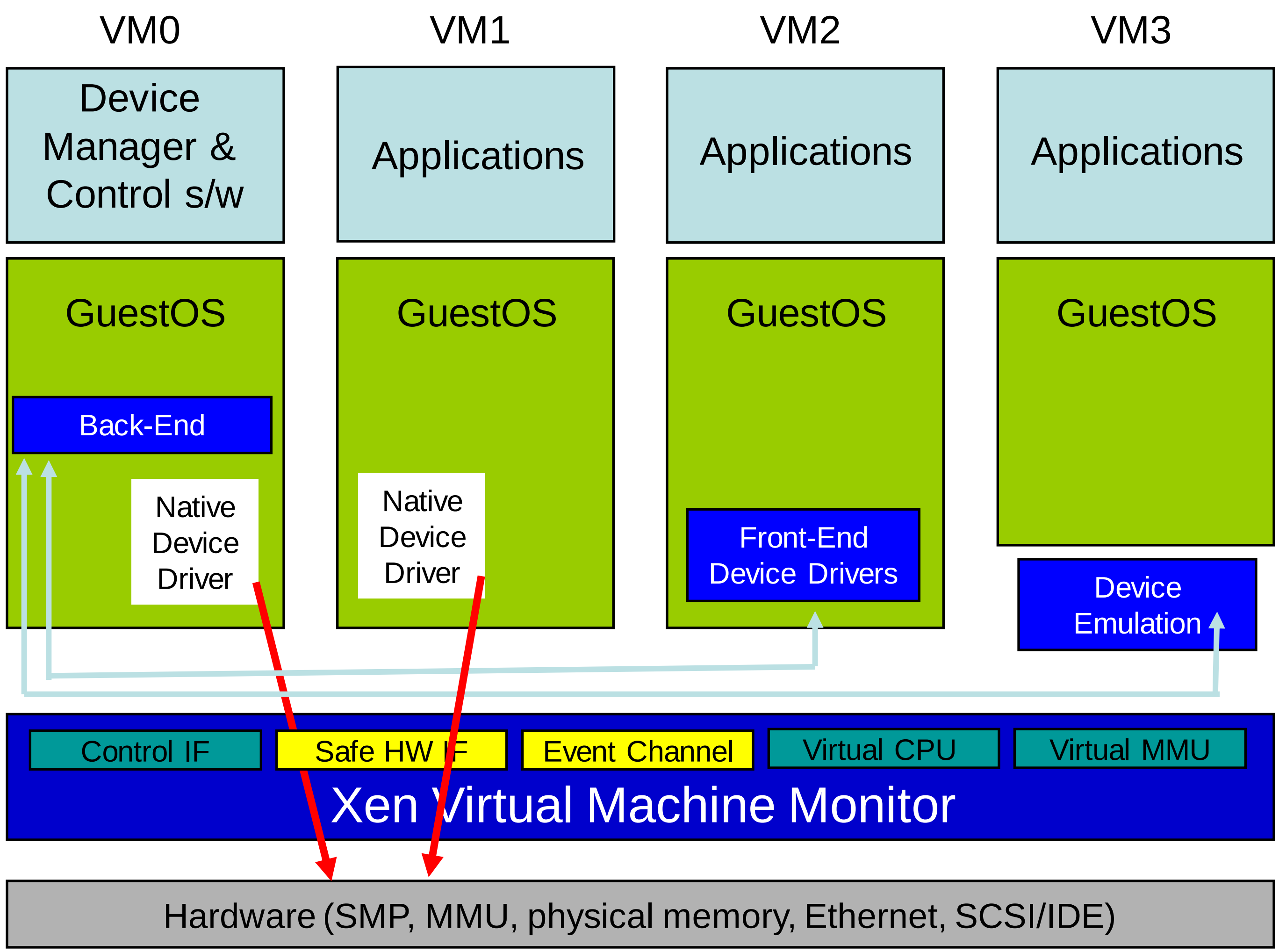
- Single free buffer, RX and TX queues
- TX and RX checksum offload
- Transmit Segmentation Offload (TSO)
- Large Receive Offload (LRO)
- Adaptive interrupt throttling
- MSI support

- (iSCSI initiator offload – export blocks to guests)
- (RDMA offload – helps live relocation)

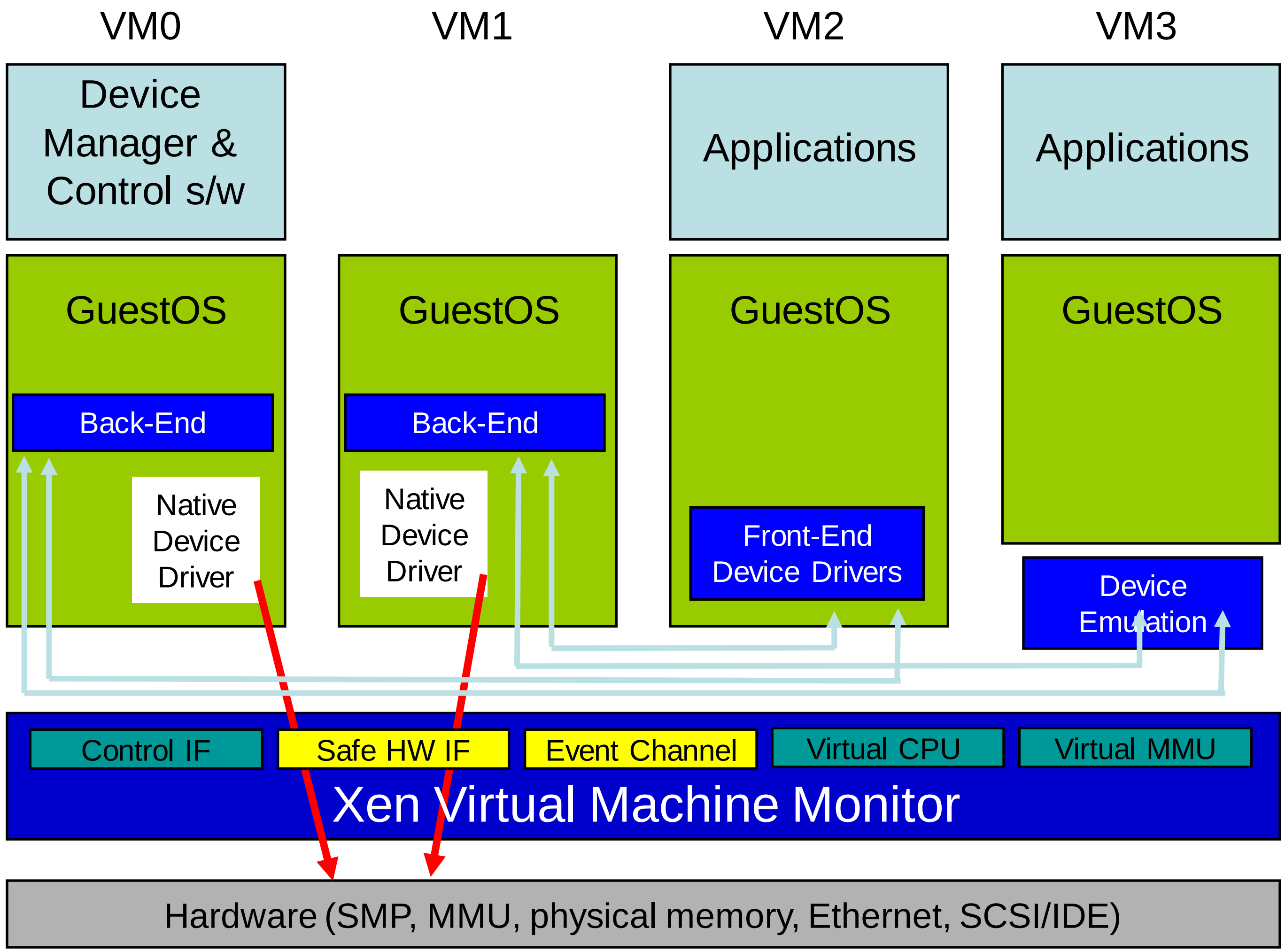
I/O Architecture



Direct Device Assignment

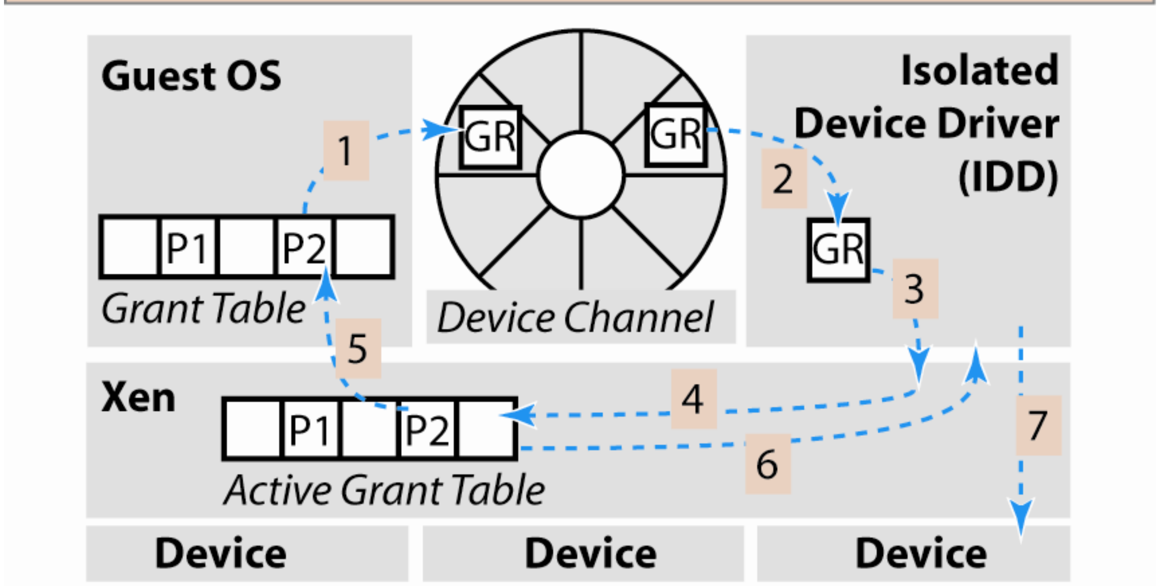


Xen3 Driver Domains



Grant Tables

Guest Requests DMA:
1. Grant Reference for Page P2 placed on device channel
2. IDD removes GR
3. Sends pin request to Xen



4. Xen looks up GR in active grant table
5. GR validated against Guest (if necessary)
6. Pinning is acknowledged to IDD
7. IDD sends DMA request to device

- **Allows pages to be shared between domains**
- **No hypercall needed by granting domain**
- **Grant_map, Grant_copy and Grant_transfer operations**
- **Signalling via event channels**

High-performance secure inter-domain communication

Level 1 : Multiple RX Queues

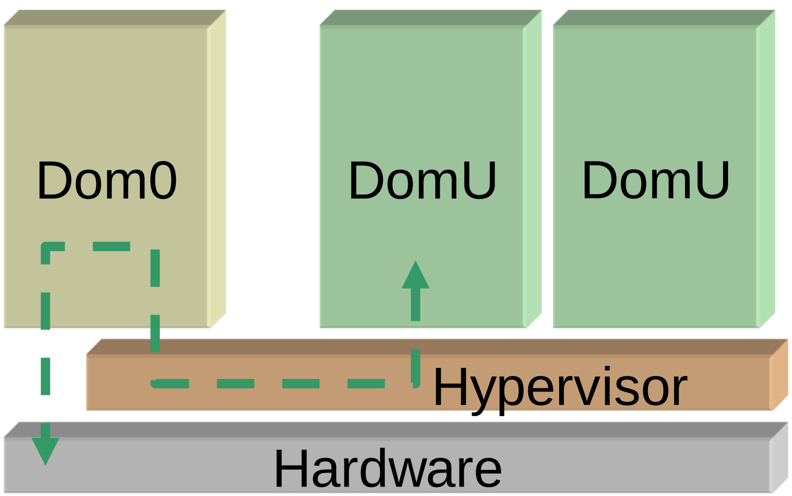
- NIC supports multiple free and RX buffer Q's
  - Choose Q based on dest MAC, VLAN
  - Default queue used for mcast/broadcast
- Great opportunity for avoiding data copy for high-throughput VMs
  - Try to allocate free buffers from buffers the guest is offering
  - Still need to worry about bcast, inter-domain etc
- Multiple TX queues with traffic shapping

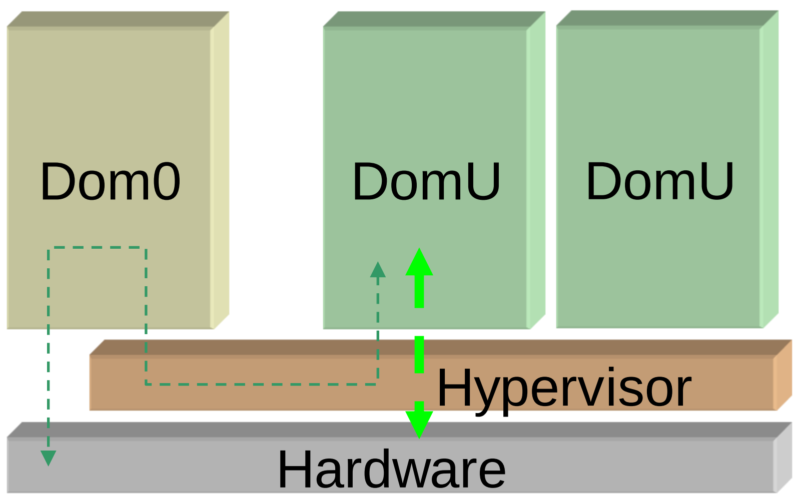
Level 2 : Direct guest access

- NIC allows Q pairs to be mapped into guest in a safe and protected manner
  - Unprivileged h/w driver in guest
  - Direct h/w access for most TX/RX operations
  - Still need to use s/w path for bcast, inter-dom
- Memory pre-registration with NIC via privileged part of driver (e.g. in dom0)
  - Or rely on architectural IOMMU in future
- For TX, require traffic shaping and basic MAC/srcIP filtering enforcement

Level 2 NICs e.g. Solarflare / Infiniband

- Accelerated routes set up by Dom0
  - Then DomU can access hardware directly
- Allow untrusted entities to access the NIC without compromising system integrity
  - Grant tables used to pin pages for DMA
- Treated as an "accelerator module" to allow easy hot plug/unplug



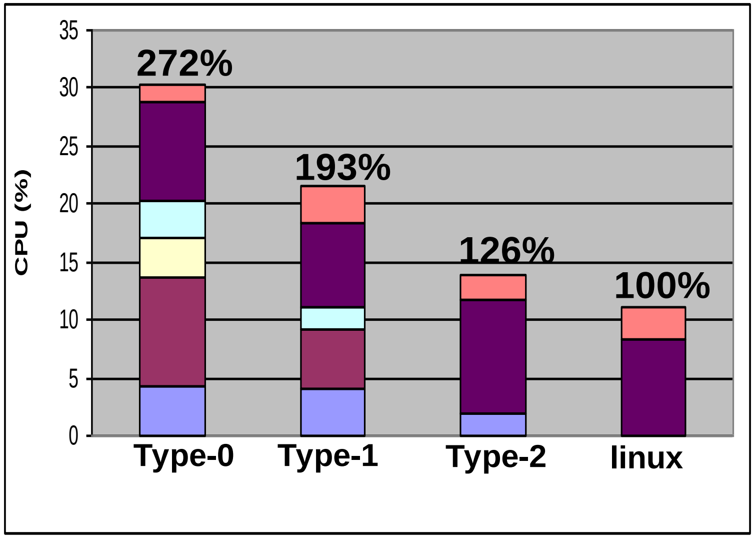


Level 3 Full Switch on NIC / MR-IOV

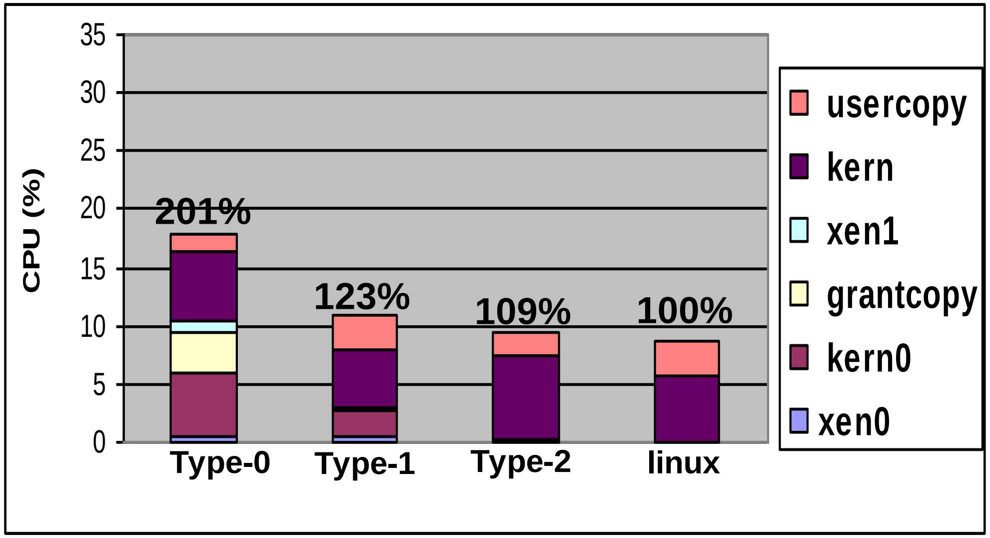
- NIC presents itself as multiple PCI devices, one per guest
  - Relies on IOMMU for protection
  - Still need to deal with the case when there are more VMs than virtual h/w NIC
  - Worse issue with h/w-specific driver in guest
- Full L2+ switch functionality on NIC
  - Inter-domain traffic can go via NIC
    - But goes over PCIe bus twice

Performance

Default configuration (6 pkt/intr)



Interrupt throttling config (64 pkt/intr)



- Smarter NICs reduce CPU overhead substantially
- Care must be taken with type-2/3 NICs to ensure benefits of VM portability and live relocation are not lost
- “Extreme late copy” for zero-copy inter-domain communication under development

Conclusions

- Open Source is a great way to get impact from University research projects
- Hypervisors will become ubiquitous, near zero overhead, built in to platform
- Virtualization may enable a new "golden age" of operating system diversity
- Virtualization is a really fun area to be working in!

ian.pratt@xen.org