Virtual Machine Monitors

Lincoln Uyeda

Virtual Machine History

- 1960s
  - IBM VM/370 - Mainframe time-sharing
- 1990s
  - VMware - MPP abstraction / x86 virtualization
  - Sun JVM – Application level virtualization

Virtual Machine History

- 2000s
  - VirtualPC - Hosted OS
  - Paravirtualization
    - Denali - 'Scalable' VM-aware network systems
    - Disco - Isolated, optimized MIPS SMP
    - Xen - x86 VMM

The Big Questions

- Why not virtualize solely at the application level?
  - Diversity of OS / ABI
  - Language requirements exclude legacy applications
- Why not virtualize across architectures as well?
  - N^2 required translators complicate VMM
- Why is virtualization useful?

Virtual Machine Motivation

- Decreasing hardware costs
  - Leads to underutilized machines
- Application isolation and security
- Legacy support
- Hardware independence
  - OS + applications become the 'machine'

VMware View of VMM Priorities

- Compatibility
  - Support for unaltered legacy OS
- Performance
  - Limit events through the VMM bottleneck
- Simplicity

CPU Virtualization Issues

- Virtualizability
  - A system is virtualizable if the VMM can retain ultimate control of the system (by running in a privileged mode)
  - Guest OS and applications must run in unprivileged mode
- Problems with x86 (IA-32) architecture
  - Instruction functionality differs depending on privileges
  - Unprivileged instructions allow access of privileged state

Techniques for CPU Virtualizability

- Paravirtualization (Disco)
  - Coupling of hardware virtualization and OS porting
  - Provide new virtualizable counterparts to the unvirtualizable instructions through the VMM
  - Port the OS to use only the virtualizable instructions

Techniques for CPU Virtualizability

- Direct execution and dynamic binary translation (VMware)
  - Trap all unvirtualizable instructions into the VMM and 'translate' them to perform the correct functionality
  - Cache translated instructions to avoid future traps

Future CPU Virtualization Trends

- Hardware support for x86 virtualization
  - Creation of a new execution mode
    - Avoids and accelerates traps for translation
    - Has the potential for direct execution VM design
    - Downside - Applications may begin using this execution mode themselves

Memory Virtualization Techniques

- Shadow page table
  - Centralized page table managed by the VMM
  - VM updates its own page table which propagates to the shadow page table
  - VM uses the shadow page table for look-up

Memory Virtualization Techniques

- Intelligent memory reclamation
  - VMware balloon process
    - Increases 'pressure' on the VM, forcing paging
    - The assumption is that the VM has better knowledge of which pages should be paged out
  - Redundant page reclamation
    - VMM keeps track of page contents
    - Pages are merged if their content is identical
    - Copy-on-write policy employed on divergence

I/O Virtualization Techniques

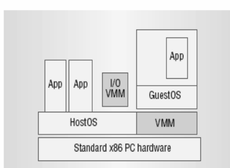
- Channel processors
  - In mainframe virtualization, separate channel processors made I/O support simple
  - Movement toward SCSI and USB based devices allows for simpler support for devices.

I/O Virtualization Techniques

- Where are the drivers?
- Two approaches
  - Hosted OS vs Hypervisor
- VMware Workstation hosted approach
  - Directs access through host OS device views and drivers
  - Introduces an expensive level of indirection

Hosted VMM Approach

Figure 2. VMware's hosted architecture. Rather than running as a layer below all other software, the hosted architecture shares the hardware with an existing operating system (HostOS).



I/O Virtualization Techniques

- Hypervisor approach
  - VMM interacts with the device and provides drivers
  - Optimized, paravirtualized devices

Hypervisor Approach

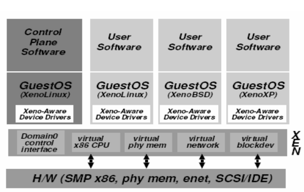


Figure 1: The structure of a machine running the Xen hypervisor, hosting a number of different guest operating systems, including *Domain0* running control software in a XenoLinux environment.

Future Trends in Virtualization

- Virtual machine migration
  - VMM level allows for encapsulation of the OS + applications
  - OS + applications can be migrated to new physical hardware while running. (VMware VMotion)
- Operating Systems as storable data
  - VM detach the hardware from the OS and applications, allowing a pure data view of the machine

Future Trends in Virtualization

- Leveraging the benefits of isolation
  - Guaranteed isolation of concurrent virtual machines allows for multiple security levels.
- Deployment via full Virtual Machines
  - Application deployment on servers requires incremental installation from OS to target applications
  - Virtual machine schemas encapsulate OS + applications into deployable templates

Xen's View of Virtualization Priorities

- Performance isolation
- High performance concurrent operation
- Compatibility of legacy applications
- Generalized VMM
  - Push architecture-specific virtualization into the actual OS (via porting)

Previous Paravirtualized Systems

- Denali isolation kernel
  - Targets thousands of virtual machines
  - Primarily focuses on virtualizing content servers
  - Alters the ABI
- Disco
  - Specific to ccNUMA machines

Control and Communication

- System management is mediated by the hypervisor, which runs in privileged ring 0
- VMM Communication
  - VMM speak to VM using asynchronous events
  - VM use synchronous hypercalls to speak to the VMM
  - Communication at this level utilizes I/O rings
    - VM can enqueue multiple requests before alerting the VMM

Abstract Data I/O Buffer Rings

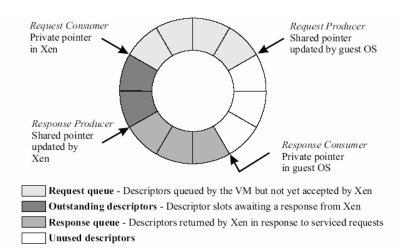


Figure 2: The structure of asynchronous I/O rings, which are used for data transfer between Xen and guest OSes.

CPU Sharing Technique

- Privileged instruction handling
  - x86 - 4 privilege levels (typically levels 1 and 2 are unused)
  - All privileged instructions are required to register and execute within Xen
  - Exception handlers, which require access to privileged state, are registered at the VMM level
  - Exception-specific optimizations
    - For system calls, fast execution handlers are allowed which do not redirect though ring 0
    - Page faults must run in ring 0, so this does not apply

CPU Sharing and Timers

- CPU scheduling
  - Borrowed virtual time scheduling algorithm
    - Developed at Stanford
    - Low-latency wake-up mechanism
    - Gives preference to recently-woken domains (VM)
- Time and timers
  - VM and VMM both have notions of time
  - Timeouts are delivered via the asynchronous events
  - Requires a switch into the VMM before delivery

CS 614 - Advanced Systems - Fall '05

Memory Management Issues

- Ideal situation
  - Tagged software TLB
    - Allows for TLB flushing of specific regions
    - VM and hypervisor can exist in separate address spaces without effecting one another
- x86 case
  - Hardware-managed untagged TLB
  - To avoid flushing with every context switch, Xen sits atop a 64 MB space at the top of every address space
  - To allocate new memory pages, the VM must register with the hypervisor VMM

CS 614 - Advanced Systems - Fall '05

Virtual Address Translation

- Full virtualization requires that the VM view physical memory as contiguous, thus it requires a shadow page table
- Xen does not attempt to provide contiguous physical memory.
  - Guest OS pages are registered with the VMM
  - When a guest OS requests an update, it is trapped and the update is validated by the VMM
  - The VMM commits all updates
  - Page frames are assigned types and reference counts to maintain access invariants and ensure VM isolation.

CS 614 - Advanced Systems - Fall '05

The Virtual View of Physical Memory

- Memory is statically partitioned between domains
- A 'balloon' driver is used to reclaim memory
- To support the sparseness of the memory, the VMM provides a single shared translation array, used by all VM

CS 614 - Advanced Systems - Fall '05

The Virtual View of Network Connections

- The VMM provides the abstraction of a firewall network router
- The VMM uses a filtering rule set and a pair of buffers for transmission and reception, as in a typical firewall router
- Guest must be able to accept packets as they arrive
  - A number of packets are provided by the VMM in exchange for a free page frame offered by the VM

CS 614 - Advanced Systems - Fall '05

The Virtual View of Disk I/O

- Disks are viewed as virtual block devices (VBD) from within domains and are accessed through I/O rings
- Disk access scheduling is optimized by reordering within the Xen VMM
- VBD appear to the guest OS much like SCSI disks
- Translation tables for each disk are maintained in the VMM

CS 614 - Advanced Systems - Fall '05

Single Domain (VM) Benchmarks

- SPEC INT2000, Linux Build Time
  - Seen as processor intensive
- Open Source Database Benchmark (Information Retrieval and Online Transaction Processing)
  - Disk Intensive
- dbench
  - Network performance on static content
- SPEC WEB99
  - Network benchmark on dynamic content

Single Domain (VM) Benchmarks

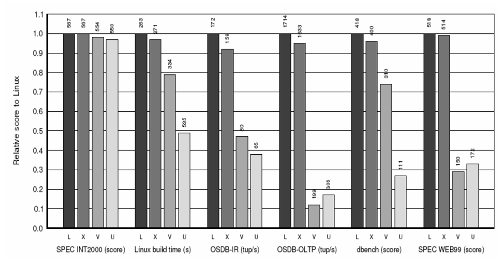


Figure 3: Relative performance of native Linux (L), XenoLinux (X), VMware workstation 3.2 (V) and User-Mode Linux (U).

Concurrent Virtual Machine Benchmarks

- Single VM benchmarking does not measure the overhead associated with concurrency support for multiple VM
- Benchmarks were performed by running multiple instances of benchmark applications on the same server as a control.
- This is compared to performance of the same benchmarking applications paired with a Xen VM for each instance

SPEC WEB99 vs. OSDB

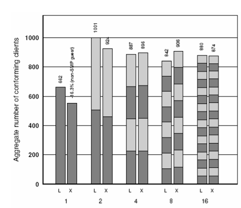


Figure 4: SPEC WEB99 for 1, 2, 4, 8 and 16 concurrent Apache servers: higher values are better.

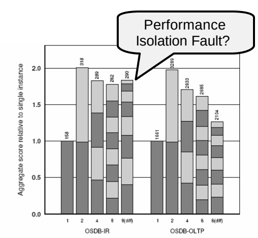


Figure 5: Performance of multiple instances of PostgreSQL running OSDB in separate Xen domains. 8(diff) bars show performance variation with different scheduler weights.

Benchmark Comments

- Xen shows predictably lower performance on benchmarks which stress page table updating and this is reflected in the results
- Isolation security was not benchmarked, though the OSDB results show significant variability in performance
- To test scalability, up to 128 VM were instantiated running SPEC CINT2000

Conclusion

- Binary translation
  - Hosted VMM
    - Simplifies hardware device access
    - Incurs the cost of indirection through the host OS
  - Hypervisor VMM
    - Allows for legacy OS support
- Paravirtualization
  - Does not require architecture specific trapping
  - Pushes the translation task into the OS code