Xen and the Art of Virtualization
Problems With True Virtualization

Insufficient architectural support
- x86 does not catch attempts to execute privileged instructions in non-privileged mode
- silent failure (i.e., no op)

Inefficiency
- attempts to write page tables when creating new processes generate a huge number of faults
- x86 does not have a software managed TLB

Implications
- explicit VMM calls are needed
- much work can be batched in a single call
Two Kinds of Virtualization

True virtualization
- no modification to guest OS source code
- good if you don’t have access to source
- performance and portability issues

Para virtualization
- needs access to OS source
- minor changes to insert hypervisor calls
- like porting OS to new architecture?
- near native performance
Xen Architecture
CPU Privilege Levels

Non-virtualized system

- Ring 3: User Apps (privileged)
- Ring 2: Unprivileged
- Ring 1: OS (privileged)
- Ring 0: Computer Hardware

Xen paravirtualization

- Ring 3: User Apps (privileged)
- Ring 2: Unprivileged
- Ring 1: Guest OS
- Ring 0: VMM
- Computer Hardware
Interfacing With Xen

Downward invocation via hypercalls
- Synchronous software trap, just like system calls

Upcalls via events
- Asynchronous event mechanism
Hypercalls

Hypercalls used for
- manipulating page tables
- allocating physical memory
- interacting with devices
Xen Memory Management

Supports architectures with hardware TLB miss handling

Xen registers guest OS page table with MMU
- page table not writeable by guest OS!
- updates done (in batches) via Xen hypercalls
- true physical memory mappings are visible!
- Xen maintains its own physical-hardware mappings for each domain
Evaluation

Relative performance
- Compared performance of Xen and two other virtualization techniques with Native Linux

Scalability
- Xen with multiple domains vs Native Linux
Relative Performance

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

Simultaneous SPEC WEB99 Instances on Linux (L) and Xen(X)
Conclusion

Xen-based Linux can achieve performance close to native Linux
Supporting many Xen-based virtual Linux machines on a single real machine has little overhead
The virtual machines are protected from each other
You can run each application in its own VM