Qumranet

evolution through convergence.

kvm:
Kernel-based Virtual Machine for Linux

Company Overview

Qumranet

- Founded 2005
- A Delaware corporation
- Locations
  - US Office – Santa Clara, CA
  - R&D - Netanya/Poleg
- Funding

SEQUOIA CAPITAL™

NVP NORWEST VENTURE PARTNERS

Expertise in enterprise infrastructure (networking, storage, servers) and virtualization

What is virtualization?

- Simulate a computer system (processor, memory, I/O) in software
- Near native performance
- Fidelity: software in a virtualized system cannot detect it is running on a virtualized system
- Examples: IBM Mainframes, VMware, Xen HVM

Uses

- Server consolidation
  - Many underutilized servers on one host
- Testing, R&D
- Virtual desktop

Virtualization basics

- Trap changes to privileged state
  - Guest cannot access hardware
- Hide privileged state
  - Guest cannot detect that the host is changing things behind its back
- Example: interrupt enable flag

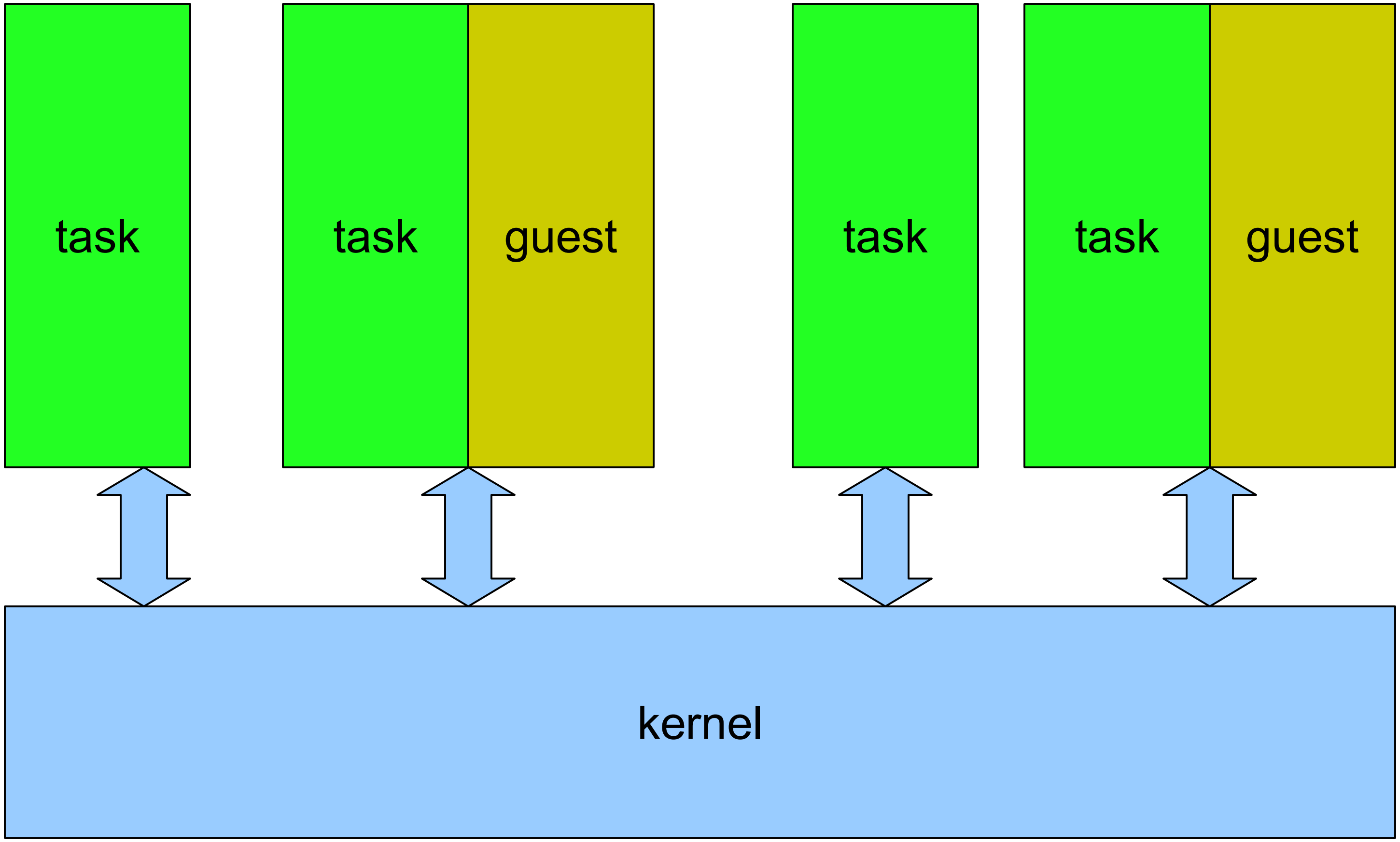
x86 hardware support

- The x86 architecture is not easily virtualizable
  - Can't easily hide some privileged state
  - VMware approach: perform just-in-time recompilation of the guest operating system
- Hardware extensions from Intel (VT), AMD (AMD-V)
  - Add additional operating modes for host and guest
  - Support for swapping state between guest and host
  - Support for hiding privileged state

kvm

- Linux kernel module exposing hardware capabilities
  - Processor state virtualization: VT
  - Memory virtualization: in kernel mode
  - I/O virtualization: mostly in userspace
- Driver kvm.ko, shows up as /dev/kvm
- Adds a third operating mode to processes: user mode, kernel mode, *guest mode*
- Zero impact on host kernel
- Open source project: http://kvm.sourceforge.net

kvm process model



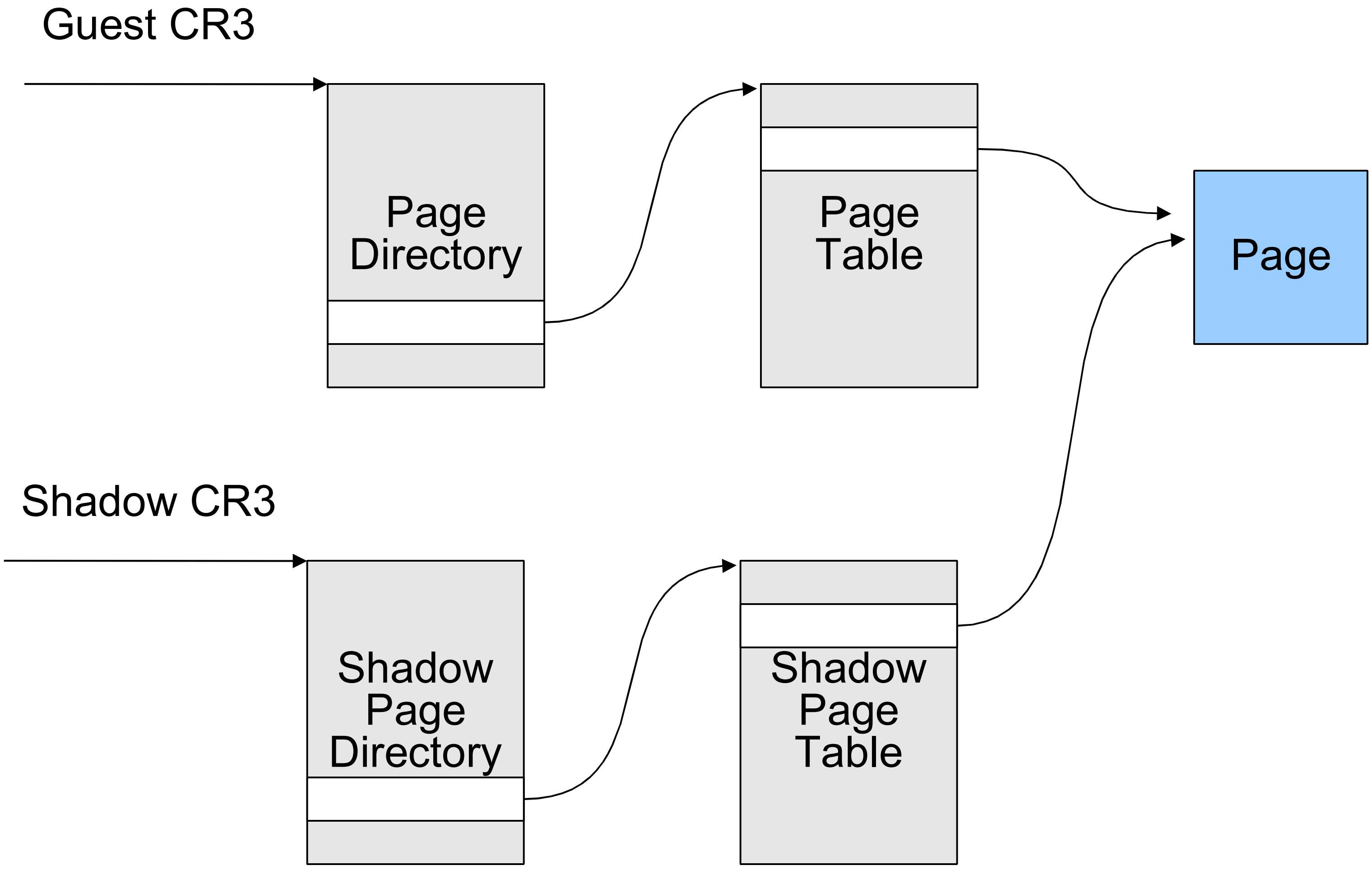
kvm process model (cont'd)

- Guests are scheduled as regular processes
- kill(1), top(1) work as expected
- Guest physical memory is mapped into the task's virtual memory space

Memory virtualization

- The processor has extensive support for translating virtual addresses to physical addresses
- When virtualizing, we need to add an additional level of translation: guest physical addresses to host physical addresses
- Solution: *shadow page tables*
  - Encode the double translation: guest virtual to host physical
  - Need to track changes to guest translations
  - Complex and expensive
- Next generation processors support multi-level translation in hardware

Memory virtualization (cont'd)



Guest CR3

Page Directory

Page Table

Page

Shadow CR3

Shadow Page Directory

Shadow Page Table

kvm vs. Xen

kvm	Xen
Part of Linux	External hypervisor
Linux scheduler, memory management	Own scheduler, memory management
Minimal impact	Intrusive
No support for paravirtualiztion	Supports paravirtualization
Under development	Fairly mature

kvm vs VMware

kvm

- Open source
- Uses VT
- Upstart

VMware

- Closed
- Uses dynamic translation
- Entrenched

Status

- Runs Windows (32-bit), Linux (32-bit and 64-bit) guests
- Intel host support published, AMD host support in the lab
- SMP hosts, uniprocessor guests
- Acceptable performance for desktops on newer processors

TODO

- Improve performance
- SMP guests

Qumranet

evolution through convergence.

thank you.

Code path examples

- Memory access
- Memory mapped I/O
- Interrupt injection

Example: memory access

- Guest accesses an unmapped memory location
- VT traps into kernel mode
- kvm walks the guest page table, determines guest physical address
- kvm performs guest physical -> host physical translation
- kvm installs shadow page table entry containing guest virtual -> host physical translation
- VT restarts execution of faulting instruction

Example: memory mapped I/O

- Guest accesses device register
- VT traps into kernel mode
- kvm determines that access is to a virtualized device
- kvm feeds faulting instruction into an internal x86 emulator to determine exact operation
- kvm exits to userspace to service the I/O
- Userspace device emulator emulates the access
- Userspace returns to kvm
- kvm returns to guest mode, after faulting instruction

Example: interrupt injection

- I/O operation completes in userspace
- Emulated device injects interrupt through kvm
- kvm sets up VT registers to inject interrupt
- Next transition to guest mode will inject a virtual interrupt