Developments in KVM on Power
Outline

- Introduction
- Little-endian support
- OpenStack
- Nested virtualization
- Guest hotplug
- Hardware error detection and recovery
Introduction

- **We will be releasing POWER® machines with KVM**
  - Announcement by Arvind Krishna, IBM executive

- **POWER8® processor disclosed at Hot Chips conference**
  - 12 cores per chip, 8 threads per core
  - 96kB L1 cache, 512kB L2 cache, 8MB L3 cache per core on chip
Introduction

• “Sapphire” firmware being developed for these machines
  – Team led by Ben Herrenschmidt
  – Successor to OPAL

• Provides initialization and boot services for host OS
  – Load first-stage Linux kernel from flash
  – Probe the machine and set up device tree
  – Petitboot bootloader to load and run the host kernel (via kexec)

• Provides low-level run-time services to host kernel
  – Communication with the service processor (FSP)
    • Console
    • Power and reboot control
    • Non-volatile memory
    • Time of day clock
    • Error logging facilities
  – Some low-level error detection and recovery services
Little-endian Support

• Modern POWER CPUs have a little-endian mode
  – Instructions and multi-byte data operands interpreted in little-endian byte order
  – Lowest-numbered byte is least significant, rather than most significant
  – “True” little endian, not address swizzling as on old 32-bit PowerPC processors

• Enabled by an MSR (machine state register) bit
  – Hypervisor register controls MSR[LE] setting on interrupt delivery

• Little-endian mode has little or no performance impact
  – Some misaligned loads/stores trap on older processors (POWER6, POWER7)

• Growing interest in running entire OS in little-endian mode
  – Ease porting of programs from other architectures
  – Ease porting of programs which access files containing LE binary data
  – Ease communication with GPUs

• New OpenPower Consortium
  – IBM, Google, Tyan, Nvidia, Mellanox

• Want to be able to run little-endian OS as KVM guest
  – Host-side changes surprisingly minor
  – Host always big-endian for now
Little-endian Support

- “Bi-endian” support – KVM guests can switch endianness at will
  - Current execution mode under direct guest control
  - Interrupt delivery mode controlled via new H_SET_MODE hypercall
- PAPR paravirtualization interface is explicitly big-endian
  - Memory operands for PAPR hypercalls are big-endian, therefore need to be byte-swapped by LE guest kernels
  - Values in registers don't need byte swapping: registers don't have endianness
  - Memory areas shared between host and guest (Virtual Processor Areas) remain BE
- Instruction emulation requires byte-swapping by KVM
  - Only occurs for MMIO emulation
  - Byte-swap instructions after reading them from the guest
  - Byte-swap multi-byte data values for normal load/stores, not for byte-reversing loads/stores
- Virtio data structures are in guest endian order
  - New virtio specification will specify little-endian
  - For current guests, QEMU and KVM have to byte-swap for little-endian guests
  - Guest endian mode sampled at virtio device reset time
Little-endian Support

- **Guests start out in big-endian mode**
  - Revert to big-endian on reboot

- **SLOF (guest boot firmware) runs in big-endian mode**
  - Will be modified to be able to load both BE and LE images

- **LE kernels check current mode, switch to LE if necessary**
  - Uses instruction that is no-op in LE mode, branch in BE mode
    - $48\ 00\ 00\ 0c\ b\ .+12$
    - $0c\ 00\ 00\ 48\ twi\ 0,r0,72$ (trap never)
  - Set MSR[LE] and do H_SET_MODE if necessary

- **No difference between how LE guests and BE guests are started**

- **Choice of LE vs. BE is a question of what image gets deployed in the guest**
  - Cataloguing problem at the same level as choice of distro
  - All the same architecture as far as libvirt and management tools are concerned.

- **POWER8 adds split little-endian mode**
  - Allows instruction and data endianness to be different
OpenStack

- OpenStack is important as management stack for KVM on Power machines
- Upstream unmodified OpenStack can now manage Power compute nodes with KVM
  - Necessary fixes are upstream
    - libvirt: some x86-centric assumptions
    - libguestfs: bug in partition table parsing
  - May need extensions to include LE/BE indication in image catalogs

- **Requirement for nested virtualization**
  - Needed to participate in OpenStack's continuous integration process

- **Requirement for guest PCI hotplug**
  - Virtual disk and network adapters
Nested Virtualization

- **OpenStack CI tests proposed patches in virtual cluster**
  - Compute nodes of virtual cluster need to be able to run guests
  - Nodes are KVM guests, therefore don't have access to hypervisor mode
  - Two options: full emulation, or “PR” style KVM
    - PR KVM, developed by Alex Graf, runs the guest entirely in user mode (“PR”oblem state) and emulates all privileged instructions and the MMU

- **Full emulation has problems**
  - Very slow
  - QEMU does not implement all the instructions in POWER6/7/8
  - Some Linux distributions provide packages optimized for POWER7
    - Fedora .ppc64p7.rpm packages since Fedora 18

- **PR KVM is our proposed solution for nested virtualization**
  - Not as fast as “HV” style KVM, but a lot faster than full emulation
  - Doesn't currently support all the features of Power processors
    - Data breakpoint (watchpoint) support
    - Performance monitor unit
    - New POWER8 features such as transactional memory
    - Supporting these features is a matter of coding
  - Not currently possible to compile both PR and HV KVM in one kernel
Nested Virtualization

- **Want to make PR and HV KVM both available in one kernel**
  - Distros won't make two kernel builds available, so will pick one or the other

- **Neither is a superset of the other**
  - HV is faster than PR, assuming necessary hardware support is available
  - HV KVM requires a paravirtualized guest kernel
    - Hardware not designed to support full virtualization; guest access to hypervisor facilities traps to the guest, not the host
  - HV KVM doesn't support emulation of ancient, embedded or 32-bit processors
    - Hardware compatibility mode for emulation of POWER6 and POWER7

- **My proposal from early August:**
  - Modify both PR and HV so that both can be compiled into one kernel
  - Each VM has an associated type: PR, HV or unknown
  - Change type to HV when PAPR capability enabled (if hardware is capable)
  - Change type to PR when first vcpu is run otherwise
  - Some problems/objections
    - Users might unexpectedly get lower-performance option than they expected

- **Aneesh Kumar's patches (early October)**
  - Split module into three: HV, PR and core
  - Userspace chooses type at VM creation time
Guest PCI Hotplug

• **Primarily for virtio devices rather than real PCI adapters**
  – Virtio devices appear as emulated PCI adapters
  – OpenStack typically boots guests with minimal configuration and adds disks and network adapters with hotplug

• **PAPR includes architecture for hotplug**
  – All sorts of resources: CPUs, memory, PCI devices, PCI host bridges
  – Referred to as Dynamic Logical Partitioning (DLPAR)
  – Designed for PowerVM environment
    • Operation initiated from management console, not the guest
    • Proprietary closed-source daemon in the guest, talking via socket to management console using proprietary protocol
    • Daemon performs necessary firmware and system calls

• **Existing guest OSes don't automatically have support for hotplug**
  – Even if they do include the proprietary daemon, we can't and don't want to use it

• **Alternative approach being developed**
  – Extend existing open-source event logging daemon (rtas_errd)
  – Define new events indicating addition/removal of PCI adapters
  – Modify QEMU to generate these events and handle resulting RTAS firmware calls (patches being developed by Mike Roth, Mike Day and Nathan Fontenot)
Hardware Error Detection and Recovery

• **Exploit Reliability, Availability and Serviceability (RAS) features of the hardware**
  - Hardware has a lot of error checking and recovery facilities
  - Parity or ECC on almost everything
  - Micro-checkpointing of the core, rollback on transient errors
  - Don't have PowerVM to provide software support

• **Error detection**
  - CPU-generated Machine Check interrupt
    • Use of data with uncorrectable errors
    • Access to non-responsive physical address
    • Parity errors in SLB or TLB
    • Duplicate SLB entries (can be generated by guest)
  - CPU-generated Hypervisor Maintenance interrupt
  - FSP scans for other transient, corrected errors and generates event logs
  - Enhanced Error Handling (EEH) in PCI host bridges
    • Isolates PCI adapters when error detected to prevent propagation of bad data
    • Errors include attempts to access outside of permitted bus address range as well as parity errors and timeouts
Hardware Error Detection and Recovery

• **Host machine check handler**
  - Patches posted by Mahesh Salgaonkar
  - Attempt to correct MMU-related errors in real mode
    - Potentially still in guest MMU context at this point
  - Then transfers to guest exit code if the machine check occurred while in a KVM guest
    - KVM has to deliver a machine check to the guest in this case since SRR0/1 registers may have been live
  - For use of data with uncorrected data, exploit hwpoison infrastructure

• **EEH support for PCI pass-through to guests**
  - EEH isolation events can be caused by guest mis-programming of adapter, or adapter failure
  - Need to notify guest of event via RTAS event-log infrastructure as specified in PAPR
  - Need to implement RTAS firmware calls to reset and de-isolate adapter

• **Other host-side RAS features don't impact KVM**
  - Daemon/database for logging and retrieving errors and other events
  - Host platform dumps
  - System catalog/VPD tools
  - Firmware update tools – system, FSP, I/O adapters
Legal Statement

This work represents the view of the author and does not necessarily represent the view of IBM.

IBM, IBM (logo), AIX, POWER, POWER6, POWER7, POWER8 and PowerVM are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.