



IBM

# Virtual-To-Emulation (V2E): Real-Mode Support

---

September 2006

*Leendert van Doorn, IBM Research*  
*Khoa Huynh, IBM Linux Technology Center*

# Agenda

---

- Goals and References
- Real-mode support with V2E
- Current Status
- Possible Future Work

# Project Goals

---

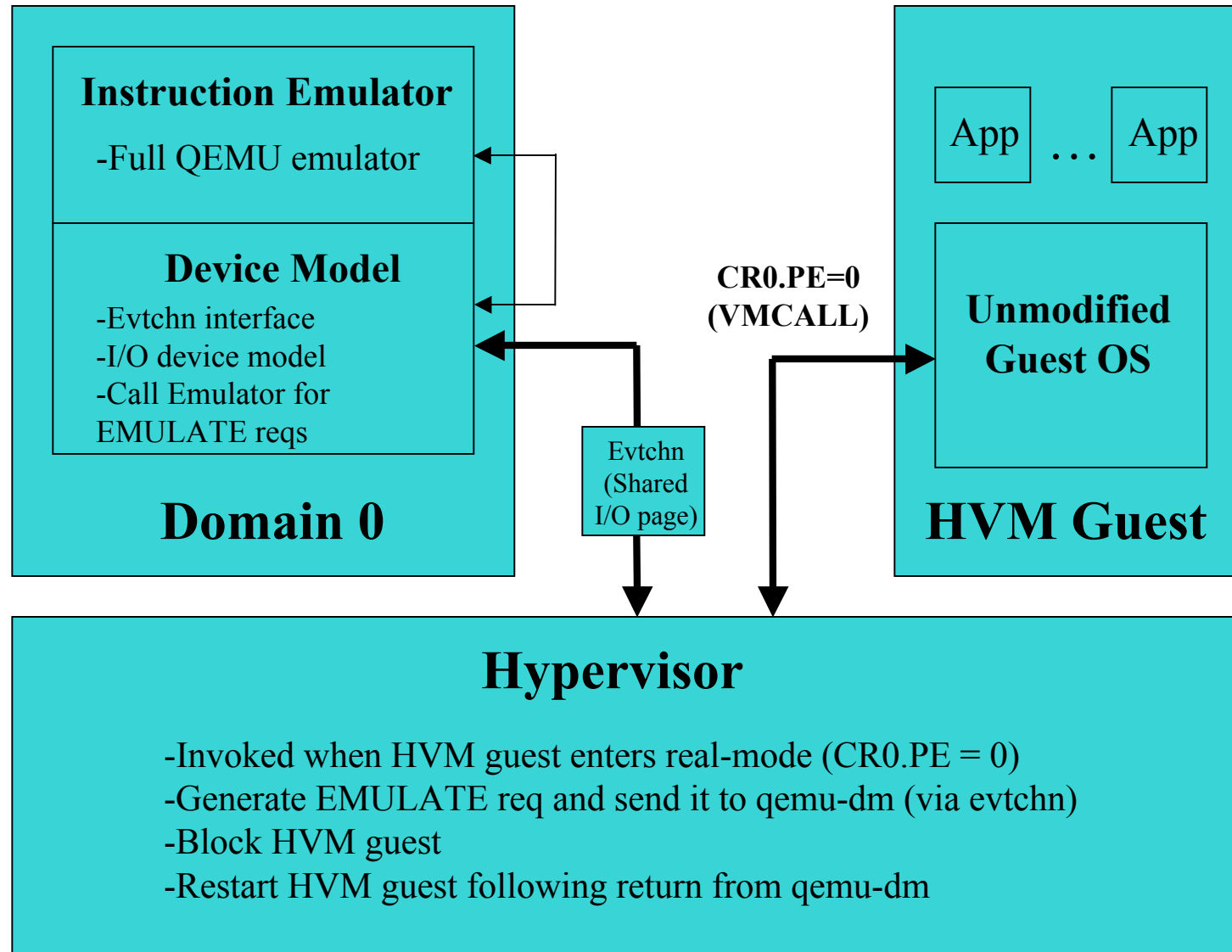
- Goals:
  - Use the concept of “demand emulation” to provide real-mode support for HVM (unmodified) guests
    - Switch to a full instruction emulator (running in domain 0 inside qemu-dm) for real-mode support
    - Switch back to HVM guest when in protect mode
  - A similar concept of “demand emulation” was used for tracking tainted code as described in a Eurosys 2006 paper
    - Practical Taint-Based Protection using Demand Emulation
      - Authors: Alex Ho, Michael Fetterman, Christopher Clark, Andrew Warfield, and Steven Hand
  - This work replaces the existing VMXASSIST code in the hypervisor

# V2E Overview

---

- Adding QEMU instruction emulation code back into QEMU device model (qemu-dm)
- Communication between Xen hypervisor and qemu-dm
  - Transfer HVM context between Xen hypervisor and instruction emulator in qemu-dm
- Criteria for switching between Xen hypervisor and qemu-dm

# V2E: Real-Mode Support



# QEMU Instruction Emulator

---

- Merge code in ioemu/target-i386 into ioemu/target-i386-dm
  - Most of the merged code is for handling processor state and instruction emulator initialization (e.g. cpu.h, helper2.c)
  - Micro operations for i386 are symbolically linked from target-i386 into target-i386-dm
- Modify Makefiles to compile instruction emulator code with existing device model
  - Instruction emulator code requires gcc 3.x
  - On distros with gcc 4.x, compatibility gcc 3.x package is required
    - Makefiles & configure files modified to accommodate this

# Hypervisor & qemu-dm

---

- HVM guest enters real mode (CR0.PE is changed to 0 and other Conditions are Met)
  - Hvmloader vmcall  
VMX\_VMCALL\_RESET\_TO\_REALMODE
  - Hypervisor is entered (via VM Exit)
    - Generate an upcall into qemu-dm via event channel
      - Create new I/O request type: EMULATE
      - Use shared I/O page (vcpu\_iodata) to contain HVM context
      - Send the EMULATE request to qemu-dm
    - HVM guest is blocked (hvm\_wait\_io())

# Hypervisor & qemu-dm

---

- QEMU-DM receives EMULATE request from hypervisor
  - EMULATE-type I/O request delivered via event channel
  - Retrieve HVM context from shared I/O page (vcpu\_iodata)
  - Put HVM context into instruction emulator
  - Call the main instruction emulator loop to invoke the instruction emulator (cpu\_exec())
  - Instruction emulator in qemu-dm starts emulating code for HVM guest (i.e., real-mode code and more)

# Hypervisor & qemu-dm

---

- When do we exit instruction emulator in qemu-dm and return to hypervisor ?
  - After executing a translation basic block, the emulator checks to see if
    - Any of the segment descriptors are in real-mode (vm8086) ? If so, continue with the next basic block
    - If not, exit the emulator if more than 1,000 basic blocks have been executed
  - After exiting instruction emulator loop
    - Save HVM context in shared I/O page (vcpu\_iodata)
    - Generate an event to return to the hypervisor (via event channel)
- Hypervisor restarts HVM guest

# Current Status

---

- 32-bit Linux guests
  - SLES9 SP2 and RHEL3 U5 boot OK
  - gfxboot (real-mode) works
    - gfxboot did not work with existing VMXASSIST code
  - LTP test suite completes successfully
- 64-bit Linux guests
  - 64-bit port is under way
- 32-bit Windows guests
  - Debugging is under way
  - Need to synchronize device states between hypervisor and instruction emulator in qemu-dm for devices emulated in hypervisor

# Current Status (cont'd)

---

- Current target: Xen 3.0.4
- Current code can be obtained from <http://xenbits.xensource.com/ext/xen-unstable-hvm.hg>
- Merging with latest xen-unstable tree is under way

# Possible Future Work

---

- Extend this work to I/O emulation
  - Improve efficiency since cost of up-calls into qemu-dm is high
  - Need to tune switching criteria between qemu-dm instruction emulator and hypervisor
  - Need to consider SMP issues (e.g. one thread running on emulator while other threads running in HVM guest)
  - Remove all redundant emulation code in hypervisor