Extending KVM with new Intel® Virtualization technology

2008. 06. 12

Sheng Yang
sheng.yang@intel.com
KVM Forum 2008
Agenda

• Accelerating MMU with Intel® Extended Page Table
• Managing TLB with Intel® Virtual Processor Identification
• Boosting Windows performance with Intel® FlexPriority
Guest can have full control over Intel® 64 page tables / events

- CR3, INVLPG, page fault

VMM controls Extended Page Tables

CPU uses both tables

EPT (optionally) activated on VM entry

- When EPT active, EPT base pointer (loaded on VM entry from VMCS) points to extended page tables
- EPT deactivated on VM exit
KVM EPT design policy

• Reuse shadow code
  – Non-paging shadow convert from Guest Linear address to hpa
    • Guest Linear Address = gpa in real mode
  – EPT convert from gpa to hpa
  – So, EPT can reuse same logic with non-paging shadow code with minor changes
    • Trap through EPT violation instead of page fault
    • Merge EPT entry with shadow entry

• Build EPT dynamically
  – Set it up using EPT Violation VM Exit
Filling EPT entry when EPT violate

- **EPT**: Extended Page Table
- **Virtual MMU**: Virtual Memory Management Unit
- **Hardware EPT TLB**: Hardware's Extended Page Table Translation Lookaside Buffer
- **Guest Process**
- **Guest Kernel**
- **CR3**: Page Table Base Register
- **Linux**
- **KVM**
- **EPT**

**Flow Diagram**:
1. Look up EPT TLB
2. EPT Violation VM Exit
3. VM Entry to Guest

**Legend**:
- **Guest’s sight**
- **Hardware’s sight**
- **EPT flow**
- **Existed entry**
- **To-be filled entry**
• Memory-mapped IO accessing
  – Host physical address don’t indicate a memory region

• Dirty logging
  – Mark dirty and reset read only EPT entries

• EPT building
Further work

- Memory over-commit
  - Swapping with MMU notifier

- Memory Type support
Agenda

- Accelerating MMU with Intel® Extended Page Table
- Managing TLB with Intel® Virtual Processor Identification
- Boosting Windows performance with Intel® FlexPriority
VPID: Motivation

- First generation of Intel® VT forces flush of Translation Lookaside Buffer (TLB) on each VMX transition

- Performance loss on all VM exits

- Performance loss on most VM entries
  - Most of the time, the VMM has not modified the guest page tables and does not require TLB flushing to occur
  - Exceptions include emulating MOV CR3, MOV CR4, INVLPG
  - Better VMM software control of TLB flushes is beneficial
VPID Basics

- VPID activated if new “enable VPID” control bit is set in VMCS

- New 16-bit virtual-processor-ID field (VPID) field in VMCS
  - VMM allocates unique value for each guest OS
  - VMM uses VPID of 0x0000, no guest can have this VPID

- Cached linear translations are tagged with VPID value

- No flush of TLBs on VM entry or VM exit if VPID active
**VPID in KVM**

- **VPID management**
  - Host (all pCPUs) uses VPID 0x0000
  - Each guest VCPU get a unique VPID
  - Recycle VPID when the VCPUs are freed
  - A bitmap is used for up to 64k VPIDs
    - *First come, first serve*

- **INVVPID for necessity**
  - VCPU migrating to a new pCPU
    - *Including VCPU creating*
  - For shadow
    - VMM modified guest page table
    - Guest CR0, CR3, CR4 changed
Agenda

• Accelerating MMU with Intel® Extended Page Table
• Managing TLB with Intel® Virtual Processor Identification
• Boosting Windows performance with Intel® FlexPriority
Task Priority Register

- TPR is accessed very frequently by 32bit Windows XP and 2003

- Previous handling in KVM
  - Trap and emulate
  - Consume a lot of CPU cycles

- But most VM Exit is not needed
  - TPR read
    - No VM Exit needed
  - TPR write
    - No VM Exit needed when increasing TPR
    - No VM Exit when decreasing TPR unless lower interrupt can be injected
Accelerating TPR accessing

- **Binary patching in KVM**
  - Replace the TPR accessing instruction with direct memory read/write instruction
  - Only support one VCPU till now

- **Intel® FlexPriority**
  - CPU maintains a “virtual TPR” register in memory
FlexPriority Work Flow

- Guest Process
- Guest Kernel
- APIC Page
- Virtual APIC Page
- TPR Register
- Virtual TPR Register
- Update TPR threshold VM Exit
- VMX
- Linux
- KVM
FlexPriority Detail

• **APIC-access Page and Virtual APIC Page**
  – Two 64bit fields in VMCS
  – APIC-access Page contains host physical address of guest APIC page
  – Virtual APIC Page contains host physical address of the page stored the LAPIC registers.

• **Accessing APIC-access Page**
  – If the offset is 0x80(TPR), FlexPriority works
  – Otherwise a APIC-access VM Exit occurred

• **Determine when should VM Exit occur when accessing TPR**
  – TPR Threshold
    • The smaller one of current TPR and maximum IRR
Legal Information

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

- Intel may make changes to specifications, product descriptions, and plans at any time, without notice.

- All dates provided are subject to change without notice.

- Intel is a trademark of Intel Corporation in the U.S. and other countries.

- *Other names and brands may be claimed as the property of others.

- Copyright © 2007, Intel Corporation. All rights are protected.
Traditional MMU virtualization - shadow

- **Defects**
  - Complex
  - Lots of VM Exits
  - Bad scalability
Paging Mode switching in EPT

- No real mode in EPT
- Set up identity mapping page table for nonpaging mode at first.
- Update KVM recorded CR3 on VM Exit