A KVM friendly IOMMU API for Linux

Operating System Research Center
Joerg Roedel

August 30, 2007

Outline

AMD
Smarter Choice

1)The current state of IOMMU support

2)Features of an IOMMU

3)The Problems with the current state

4)General Design of the proposed API

5)Address Mapping Functions

6)Pagetable Sharing with NPT/EPT

7)Paravirtualized IOMMU

AMD
Smarter Choice

The current state of IOMMU support

The current state of IOMMU support

AMD
Smarter Choice

- Currently there exists the DMA mapping API
- Intended for devices not supporting the full host physical address range
- Provides simple functions that map host addresses to bus addresses
- Some sync functions to allow a implementation without hardware support
- Current IOMMU code only implements this API

AMD
Smarter Choice

Features of an IOMMU

Abilities of an IOMMU

AMD
Smarter Choice

- Todays IOMMU support more than simple address remapping
- Support of full 64-bit address spaces for devices
- Each device may have its own address space with protection domains
- Thus support for device isolation
- [Interrupt remapping – eliminates problems with interrupt sharing]

The Problems with the current state

The Problems with the current state

AMD Smarter Choice

- The DMA mapping API only covers a subset of the IOMMU functionality
- No explicit support for device isolation
- No possibility to describe the address space of an device explicitly
- Defining own device address space is required for device passthrough to the guest
- DMA mapping API can not be used for this purpose
- A new API is required to fit these needs

AMD
Smarter Choice

Design of the proposed API

General Design of the new API

AMD
Smarter Choice

- Support for protection domains (creation, deletion, device assignment)
- Allows defining the address space of a protection domain in 2 ways
  - map pages into the address space of the protection domain
  - use a host address space
- IOTLB management functions for the second way
- A commit function for paravirtualizing the API

General Functions

AMD
Smarter Choice

- `iommu_available()` - Checks if hardware is available
- `iommu_domain_alloc()` - Creates a new protection domain
- `iommu_domain_free()` - Deletes a protection domain
- `iommu_domain_add()` - Adds a device to a domain
- `iommu_domain_remove()` - Removes a device from a domain

Address mapping functions

- `iommu_map_page()` - Map a single page into protection domain
- `iommu_map()` - Map a number of pages
- `iommu_unmap_page()` - Unmap a page
- `iommu_unmap()` - Unmap a number of pages

Pagetable Sharing with NPT/EPT

AMD Smarter Choice

- `iommu_use_pagetable()` - Use a host pagetable
- `iommu_pt_sync_all()` - Re-read the complete host pagetable
- `iommu_pt_sync_range()` - Re-read a given region
- `iommu_pt_sync_one()` - Re-read one address

Paravirtualized IOMMU

AMD
Smarter Choice

- The API itself is easily paravirtualizable
- But usage may result in many hypercalls
- Therefore a commit function will be introduced
- Allows caching of requests to IOMMU an flush it with a single hypercall
- Function: `iommu_commit()`
- Has to be called to make all address mapping functions take affect

Trademark Attribution