Unified I/O Page Table Management for Passthrough Devices

Kevin Tian, Baolu Lu
Agenda

• A brief background
• Development plan
• Manage security context for user-initiated DMAs
• Miscellaneous opens if time allows
Proposal for a Unified Framework

* Thank Jason Gunthorpe for initiating this idea!

* Please refer to [link] for a full design proposal
Key Concept

1. **Bind**
   - **VFIO**
   - **iommufd**
   - **I/O Address Spaces**
     - address space id (ioasid)
     - type
     - page table format
     - parent ioasid (optional)
     - list of attached devices
   - **Bound Devices**
     - Struct device *(RID)*
     - PASID (optional)
     - User provided cookie

2. **DEVICE_GET_INFO**
   - **ALLOC_IOASID**
     - Map/unmap DMA
     - Invalidate iotlb
     - Fault handling

3. **Kernel-managed**
4. **User-managed**
5. **Shared**
   - (mm, ept, etc.)

---

**VFIO**

**iommufd**

**I/O Address Spaces**

**Bound Devices**

**iommu ops**

**IOMMU**

**User**

**Kernel**

**VFIO**

**Attach**
Development Plan

Basic skeleton
- vfio only
- pci only
- Type1v2 mapping semantics

v1 just sent out!

Remaining vfio devices
- SW mdev
- PPC device
- Platform device
- Device with no-snoop DMA
- Replace vfio_iommu_type1 with a shim driver (in skeleton?)

vDPA adoption

New device types
- HW mdev/subdev (PASID)

User-managed page table
- ioasid nesting (hw)
- PASID virtualization
- I/O page fault (1st-level)

- ioasid nesting (sw)
- I/O page fault (2nd-level), for on demand paging
- IOMMU dirty bit

- Shared I/O page table (mm, ept, etc.)
- Hardware-assisted viOMMU

Miscellaneous

Need community collaboration. Many tasks can be done in parallel!
Manage User-initiated DMAs

- Establish a secure context (iommu unmanaged domain) which restricts user-initiated DMAs to
  - process memory
  - sibling devices in the same group

- Guarantee exclusive DMA ownership on the group, i.e. devices in the group must be
  - Bound to the owner driver (e.g. vfio), or
  - Bound to a driver known DMA-safe (e.g. pci-stub), or
  - Driverless

- iommufd can copy what vfio does today, with one exception
  - Need manage multiple security contexts due to decoupled bind/attach

- Current IOMMU API has problem on the transition between unmanaged domains
  - Default domain is automatically re-attached after detaching from previous context

Need cooperation from IOMMU core!
Manage User-initiated DMAs (Cont.)

**iommufd (no group knowledge)**

- **Open devicefd** (block access)
  - **Bind iommufd** (unblock access)
  - **Attach ioas1**
  - **Detach ioas1**
  - **Attach ioas2**
  - **Detach ioas2**

**Device-centric IOMMU API**

- default domain
  - DMA blocked
  - user ioas1
    - DMA blocked
  - user ioas2
    - DMA blocked
  - default domain

**IOMMU Core (user-dma awareness)**

- **iommu_device_init_user_dma()**:
  - If the first device in the group
    * Validate and start monitoring group DMA ownership
      * Mark the group for user-dma
        * Block DMA for the entire group
    - Else
      * Refcount_inc(user_dma_cnt)
  - Else
    * Refcount_inc(user_dma_cnt)

- **iommu_attach_device()**:
  - If the first device in the group
    * Attach the group to ioas
    - Else
      * Refcount_inc(attach_cnt)
  - Else
    * Refcount_inc(attach_cnt)

- **iommu_detach_device()**:
  - If the last device in the group
    * Detach the group from ioas
      * Block DMA instead of re-attaching to default domain
    - Else
      * Refcount_dec(attach_cnt)
  - Else
    * Refcount_dec(attach_cnt)

- **iommu_device_exit_user_dma()**:
  - If the last device in the group
    * Clear user-dma flag
      * Re-attach the group to default domain
      * Stop monitoring group DMA ownership
    - Else
      * Refcount_dec(user_dma_cnt)
If time allows

• ioasid naming conflict
  • fd-local software handle vs. hw asid (pasid/ssid, /drivers/iommu/ioasid.c)

• Module name and devnode
  • iommufd (/dev/iommu) vs. uiommu (/dev/uoimmu)
  • Other options?

• /dev/vfio/devices/ hierarchy
  • A plain layout mixing all types together
    • /dev/vfio/devices/0000:00:14.2 (pci)
    • /dev/vfio/devices/PNP0103:00 (platform)
    • /dev/vfio/devices/83b8f4f2-509f-382f-3c1e-e6bfe0fa1001 (mdev)
  • Subdirectories based on device types
    • pci, platform, ccw, etc.
    • Pdev vs. mdev

• Do we need to build iommufd as a separated module?
• Convert vfio_iommu_type1 to a shim driver
Backup
Current Situation

VFIO

IOMMU Core

vDPA

Not a scalable architecture moving forward!
Manage User-initiated DMAs

Current VFIO

Groupfd: set container

Groupfd: unset container

Open devicefd

Close devicefd

default domain

user ioas

default domain

User-initiated DMAs

A single security context

Monitor DMA ownership