IOMMUFD Integration in QEMU

Yi Liu (Intel) & Eric Auger (Red Hat)
KVM Forum 2022, Sept 13
Agenda

- VFIO uAPI and IOMMU subsystem
- IOMMUFD Kernel Skeleton & VFIO add-ons
- QEMU VFIO Adaptation
- New Use Cases
- Conclusion
VFIO API & IOMMU Connection

/sys/kernel/iommu_groups/7
/sys/kernel/iommu_groups/7/devices
/sys/kernel/iommu_groups/7/devices/0000:05:00.1
/sys/kernel/iommu_groups/7/type
/sys/kernel/iommu_groups/7/reserved_regions

container fd (/dev/vfio/vfio)

group fd (/dev/vfio/$GROUP)

Container

Device #0

VFIO_GROUP_SET_CONTAINER
VFIO_GROUP_GET_STATUS
VFIO_GROUP_GET_DEVICE_FD

Device #1
device fd

VFIO_DEVICE_GET_REGION_INFO
VFIO_DEVICE_GET_IRQ_INFO
VFIO_DEVICE_SET_IRQS
VFIO_DEVICE_RESET

VFIO_SET_IOMMU
VFIO_IOMMU_GET_INFO
VFIO_IOMMU_MAP_DMA
VFIO_CHECK_EXTENSION
VFIO_GET_API_VERSION
The story behind a new uAPI

• New Requirements: HW nested paging, vSVA (PASID)
  − Alloc/free PASID, Bind/unbind guest page tables, cache invalidation, fault handling, ...
  − IOMMU uAPI definition (upstreamed), tunneling through extended VFIO uAPI
• Duplicate logic in different passthrough frameworks
• Opportunity to address some vfio_iommu_type1 shortcomings
• iommufd: a new UAPI to manage IO address space pointing to user mem
IOMMUFD Kernel RFC scope

• /dev/iommu char device
• ioas, dev, hwpt (iommu_domain) object lifecycle
• auto/manual hwpt on device attach
• Complex and optimized infra for
  − IOVA alloc, mapping
  − PFN storage (xarray, iommu_domain, userspace pointer), pinning and users accounting
• IOAS shared between subsystems
• vfio container compat IOCTLs
  − Use /dev/iommu in place of /dev/vfio/vfio
• v1 does not support new use cases (nested, pasid)

---

PATCH RFC 00/12] IOMMUFD Generic interface
(Jason Gunthorpe, March 2022)
VFIO Kernel Add-ons

- Allows direct manipulation of VFIO device: `/dev/vfio/devices/vfioX` on top of legacy `/dev/vfio/$groupID`
- 2 new VFIO IOCTLs
  - VFIO uAPI
    - `VFIO_DEVICE_BIND_IOMMUFD`
    - `VFIO_DEVICE_ATTACH_IOAS`
  - uAPI
    - `/dev/iommu`
    - `ioas_id`
    - `dev_id`
    - `hwpt_id`
QEMU VFIO Adaptation [1/2]

- 2 QEMU RFC iterations: vfio: Adopt iommufd (April and June 2022)
- Adapt QEMU VFIO device to support both legacy and iommufd BE
- Split the code into IOMMU agnostic/specific
- Define an IOMMU backend interface that hides Group/Device centric handling
QEMU VFIO Adaptation [2/2]

- VFIOContainer <-> [iommufd, ioas]
- No VFIOGroup concept in iommufd
- Container base class interface and 2 derived BEs
  - vfio_attach/detach_device, check_extension, set_dirty_page_tracking, devices_all_dirtyTracking, get_dirty_bitmap, add_window/del_window, dma_map/dma_unmap, reset
- No Feature parity yet with legacy BE
Agenda

• All about IOMMU uAPI
• IOMMUFD Kernel Skeleton & VFIO add-ons
• QEMU VFIO Adaptation
• **New Use Cases**
• Conclusion
Nested Translation Recap

• **Usage**
  - Hardware-assisted vIOOMMU
    - glIOVA or vSVA

• **Vendor support**
  - Intel® VT-d, ARM* SMMUv3, etc.
  - Different architectures
    - Translation hierarchy
IOAS & HWPT

- **GPA IOAS**
  - Stores GPA->HPA mappings

- **HWPTs**
  - `auto_hwpt`: covers GPA->HPA
    - equivalent to auto-created domain in vfio container, attached by devA (ATTACH_IOAS)
  - `s2_hwpt`: covers GPA->HPA
    - vendor-specific format, attached by devB (ATTACH_HWPT)
  - `s1_hwpt`: covers gIOVA or gVA to GPA
    - vendor-specific format, attached by devB (ATTACH_HWPT)
    - nested on s2_hwpt
Nested Translation Software Architecture

- iommu driver supports allocating nested type iommu_domain, attach/detach, and cache invalidation
  - MSI doorbell handling for ARM

- IOMMUFD IOCTLS
  - IOMMUDEVICE_GET_INFO
  - IOMMU_ALLOC_USER_HWPT
  - IOMMUHWPT_INVAL_S1_CACHE

- VFIO IOCTL
  - VFIODEVICE_ATTACH_HWPT

Nested Translation in QEMU

- Nicolin Chen (Nvidia), Eric (Redhat), and I (Intel) worked together to consolidate the Intel and ARM changes into the below branch

- Stage-1 related IOMMU operation is issued in vIOMMU

- IOMMUFDDevice
  - It’s a per-device object, provides a class of callbacks like at[de]tach_hwpt
  - VFIO/vDPA is supposed to derive it and implement device-specific at[de]tach methods
  - VFIO/vDPA sets its IOMMUFDDevice object to vIOMMU per bus specific methods

- TODO
  - Long term, *we* wish to move all the iommu-related codes out of the hw/vfio folder
vSVA Recap

• Nested translation
  - Stage 1 page table is guest CPU page table

• IOMMU fault reporting
  - Recoverable fault, also known as I/O page request (PRQ)
  - Non-recoverable fault

• PASID Support
  - PASID is used to tag multiple stage-1 page tables
  - PASID virtualization differs across vendors
    - Intel® VT-d: hypervisor interception for guest PASID support
    - ARM® SMMUv3: no hypervisor interception for guest PASID
IOMMU Fault Reporting in IOMMUFD

- Lu Baolu (Intel) is moving bare-metal fault reporting to be per-domain
  - [https://lore.kernel.org/linux-iommu/20220826121141.50743-1-baolu.lu@linux.intel.com/](https://lore.kernel.org/linux-iommu/20220826121141.50743-1-baolu.lu@linux.intel.com/)

- IOCTL
  - IOMMU_PAGE_RESPONSE
  - [https://github.com/nicolinc/iommufd/commits/iommufd-v5.19-rc5](https://github.com/nicolinc/iommufd/commits/iommufd-v5.19-rc5)
PASID Virtualization for Intel® VT-d

- iommufd
  - uAPI to allocate host PASID and map guest PASID to host PASID
  - kAPI for PASID attach_hwpt
  - kAPI for querying guest PASID -> host PASID

- VFIO
  - uAPI for PASID attach_hwpt

- KVM
  - uAPI for updating VMX VMCS PASID translation table (for ENQCMD)

Still has design open on PASID virtualization
vSVA in QEMU

• Both Intel VT-d and ARM SMMUv3 emulation code is going to be updated to support iommu fault handling and PASID capability for vSVA

• Intel VT-d further requires PASID communication between device module and IOMMU emulation code
Conclusion

• IOMMUFD is a major redesign
• Significant rework at both kernel and userspace level
• Spec still unstable, especially for new features
• Feasibility of vfio_iommu_type1 deprecation is not guaranteed at this point
• Lots of kernel dependencies (cleanups, vfio/iommu code reshuffle) not merged yet
• Nicolin Chen, Lu Baolu, Eric & Yi working on prototyping nested and vSVA on ARM and Intel. Discussions need to happen to integrate other vendors (AMD).
• Other VFIO IOMMU BE such as SPAPR needs to be addressed at some point
• Other new features blocked waiting for iommufd upstream
Some References (1/2)

• Prior to iommufd
  – [RFC PATCH 0/8] Shared Virtual Memory virtualization for VT-d
  – [PATCH V4 00/18] IOASID extensions for guest SVA (Feb 21)
  – IOMMU Userspace API
  – [PATCH v7 00/16] vfio: expose virtual Shared Virtual Addressing to VMs (Sept 2020)
  – [Patch v8 00/10] vfio: expose virtual Shared Virtual Addressing to VMs (Mar/3 2021)

• iommu uAPI Discussions
  – [RFC] /dev/ioasid uAPI proposal (May 21)
  – [RFC v2] /dev/ioasid uAPI proposal (July 21)
• Post iommufd
  - RFC 00/20] Introduce /dev/iommu for userspace I/O address space management (Intel, Sept 2021)
  - [PATCH RFC v2 00/13] IOMMUFD Generic interface (Nvidia, Sep. 2022)
  - [RFC v2 00/15] vfio: Adopt iommufd
  - Kernel: https://github.com/nicolinc/iommufd/commits/iommufd-v5.19-rc5
  - Qemu: https://github.com/nicolinc/qemu/commits/qemu-iommufd-5.19-rc5

• Conferences
  - LPC 2017 Discussion
  - KVM Forum 2018: Shared Virtual Address in KVM
  - Fosdem 2019: Virtual IOMMU Implementation using HW Nested Paging
  - KVM Forum 2020: PASID Management in KVM - Yi Liu & Jacob Pan
  - LPC 2020: Enhancements to IOMMU and VFIO User APIs for guest SVA
  - LPC 2021: Unified I/O page table management for passthrough devices, in-kernel API discussion between IOMMU core and /dev/iommu