Mediated devices for ethernet
Device IO vs Device Driver

UIO/VFIO  ->  userland device driver

VFIO_MDEV  ->  userlande device IO
- Introduced in kernel 4.10.
- Currently supported by Intel i915/QEMU to support virtual GPUs.
- Offers iommu isolation using VFIO-API.

Many use cases
- WrapDrive from Huawei for accelerators (crypto...)
- net_mdev further specialized WrapDrive for net_devices
  - To be used by DPDK, ODP, VPP, Netmap, specialized stacks (TSN)
- block_mdev for SPDK...?
Design goals & roles

**Design goals**
- Userland shall not be able to highjack kernel
  - Same attack surface as VFIO
- Bus agnostic (PCI, DPAA2, platform...)
- Userland packet framework agnostic
  - VPP packet to index

**Userland**
- packet memory (buffers or areas) and HW descriptors
- Leverages netlink and other control channels

**Kernel driver**
- control of device initialization, reset, rings...
net-mdev overall architecture
Reality check

Code impact

● Realtek kernel driver: 10KLOCs

● PoC sending/receiving packets from userland
  ○ r8169/e1000e userspace-“driver”(descriptor handling/mmio): ~100 lines
  ○ r8169/e1000e full ODP integration: ~ 500 lines
  ○ Userland ODP framework: 620 lines
  ○ Kernel framework(r8169/e1000e): 227 and 239 lines respectively
  ○ r8169 driver changes: less than 80 lines
  ○ e100e driver changes: less than 50 lines

Mediated device framework to be completed

Subtle security issues solved
IOMMU/SMMU and VFIO refresher (memory)

IOMMU groups: internal

IOMMU domains: API
“Domains”: DMAR or process or VM or...
can contain multiple IOMMU groups
VFIO container

IOMMU paging
granularity

kmalloc: FFFFFFF80021000000
Physical: 20000000000
IOVA: FC000000
userland: 1000000000
Solving problems

**Finalize vfio-mdev for IOMMU**
- Currently deals only with remapping and device emulation
- Single IOVA for all groups to allow packet forwarding
- Streaming DMA handling

**Subpage PCI mapping for “door bell” and other stuff**
- Problem appears with N ports = 1 PCI device and only p<N captured ports
  - Cannot allow PCI mapping: IOCTL or CPU innovation required

**Dealing with rings setup**
- Complex activity, page aligned boundaries (MMU & IOMMU)
- Need to be domain IOVA, not group IOVA
Plan

Complete understanding of problem space

- More complete framework (statistics - TUN/TAP value?)
- Have DPDK PMD, ODP driver, VPP driver

Who wants to collaborate?

Summarize in RFC for upstream and get guidance