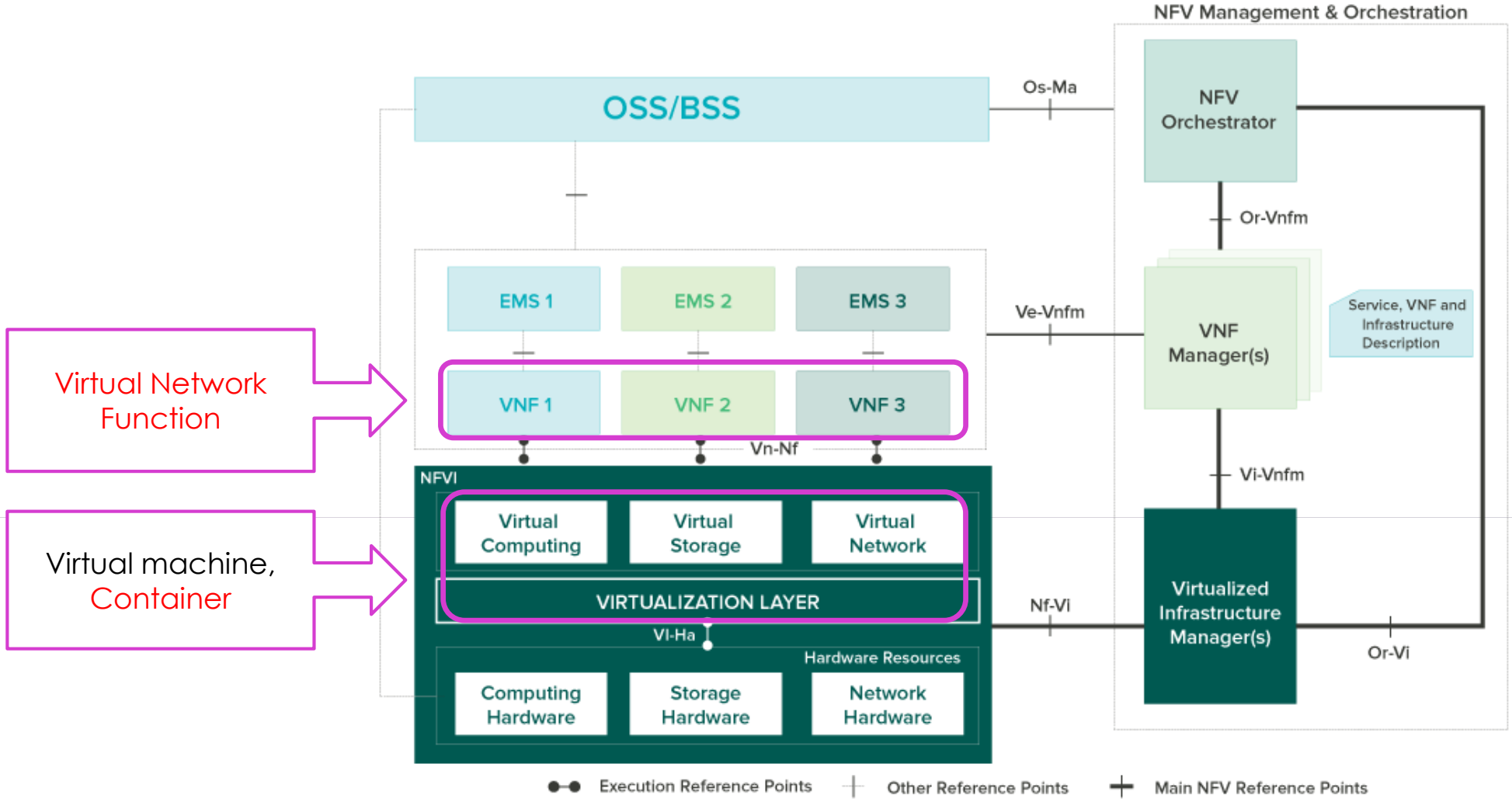

DPDK
Scalable High-Performance User Space Networking for Containers
Cunming Liang, Jianfeng Tan - Intel
DPDK US Summit - San Jose - 2016

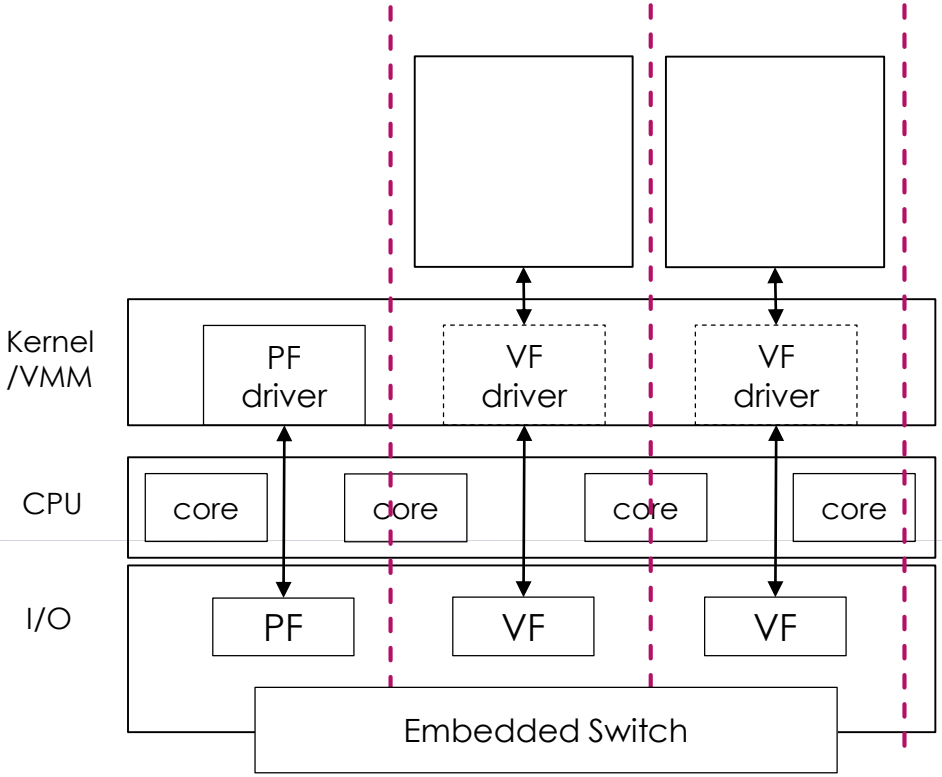
Outline

- Container-based VNF, why DPDK?
- Accelerate Network I/O for Container
- Be More Friendly to Container
- Future work

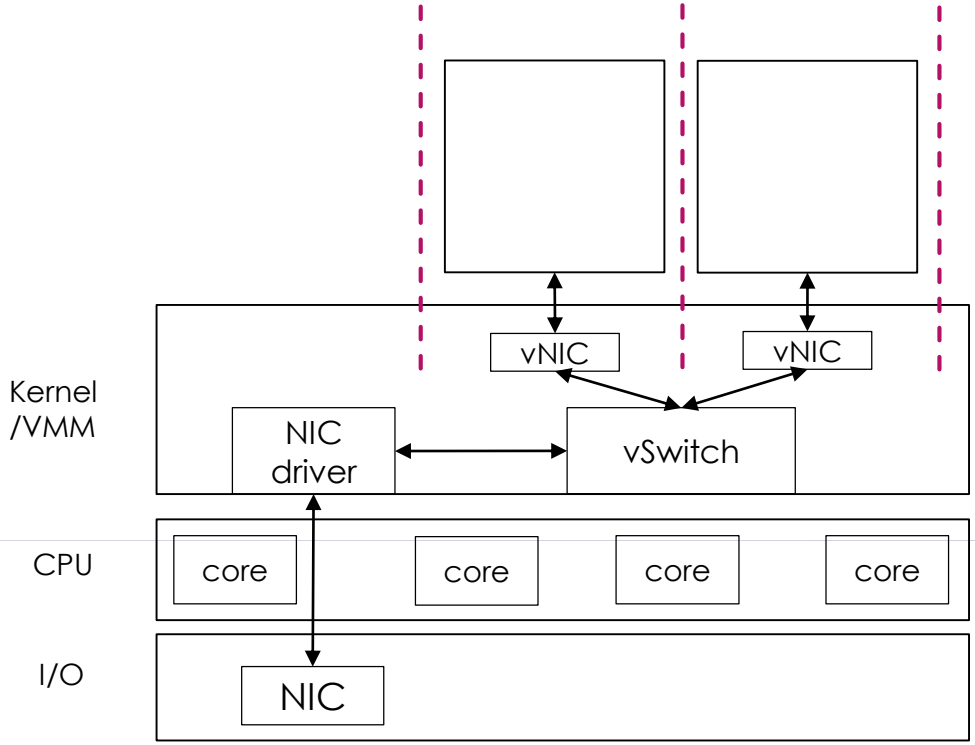
NFV and Container



I/O Virtualization Model for NFVi



Slicing

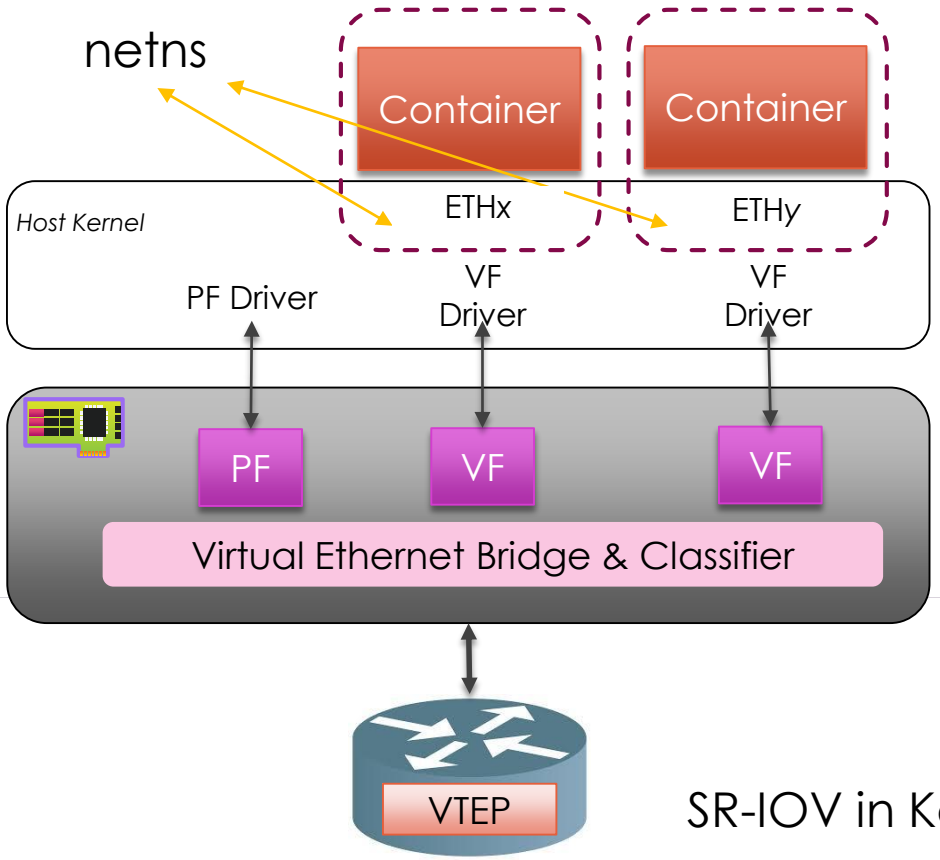


Aggregation

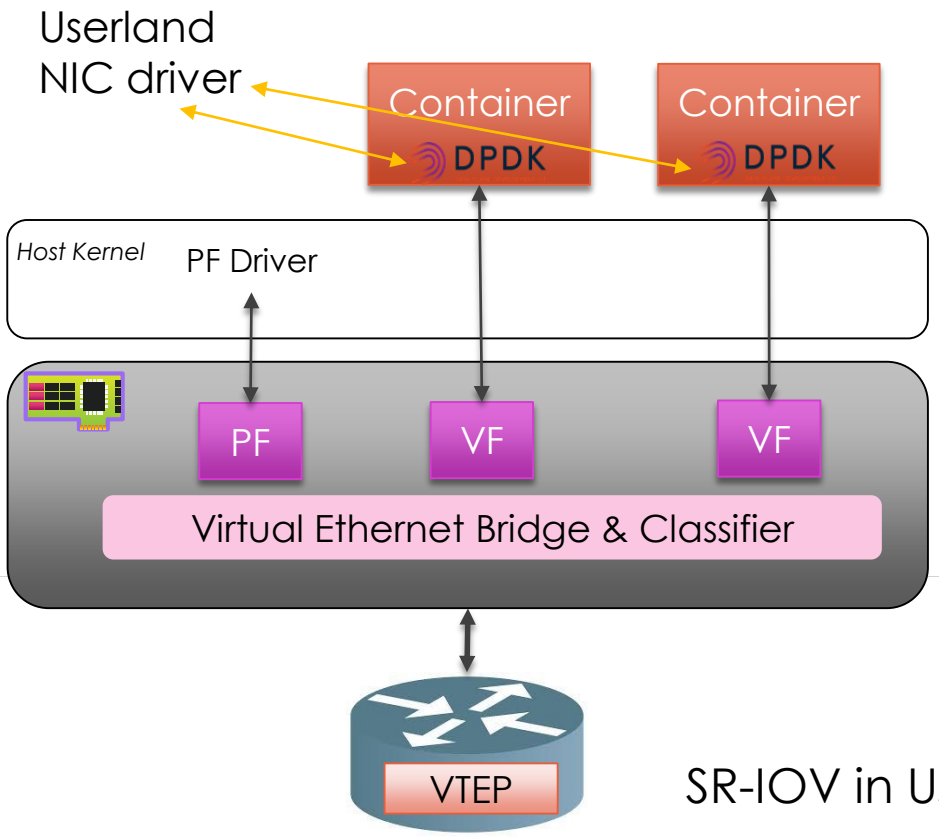
Accelerate Container-based VNF

- VNFs
  - LB, FW, IDS/IPS, DPI, VPN, pktgen, Proxy, AppFilter, etc
- Benefits
  - Provisioning time - SHORT
  - Runtime performance overhead - LOW
- Challenges
  - Security/Isolation
  - **High performance networking**
    - High throughput
    - Low latency
    - Jitter (deterministic)

DPDK SR-IOV PMD for Container



SR-IOV in Kernel



SR-IOV in Userland

Setup Userland SR-IOV with DPDK

- Prepare VFs

```
$ echo 1 > /sys/bus/pci/devices/0000\:81\:00.0/sriov_numvfs
$ ./tools/dpdk_nic_bind.py --status
…
0000:81:00.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection' if=eth1
drv=ixgbe unused=
0000:81:10.0 '82599 Ethernet Controller Virtual Function' if=eth5 drv=ixgbevf
unused=
…
```

- Bind to vfio driver

```
$ modprobe vfio-pci
$ ./tools/dpdk_nic_bind.py -b vfio-pci 0000:81:10.0
```

- Prepare hugetlbfs

```
$ mount -t hugetlbfs -o pagesize=2M,size=1024M none /mnt/huge_c0/
```

- Start container

```
$ docker run … -v /dev/vfio/vfio0:/dev/vfio/vfio0 -v
/mnt/huge_c0/:/dev/hugepages/ …
```

http://developers.redhat.com/blog/2015/06/02/can-you-run-intels-data-plane-development-kit-dpdk-in-a-docker-container-yep/

Deterministic Environment (1)

- Deterministic CPU env
  - Boot-time: disable timer / task scheduler
    - … default_hugepagesz=1G isolcpus=16-19 …
    - Reducing scheduling-clock ticks: *adaptive-tick* mode
  - Run-time: core-thread affinity
    - cpuset tool: taskset / numactl
    - cgroup.cpuset: cset / docker run … --cpuset-cpus …
  - BIOS setting: if necessary, disable *Hyper-Threading*

Deterministic Environment (2)

- Deterministic cache env
  - Data Direct I/O (DDIO) technology
  - Cache Allocation Technology (CAT)

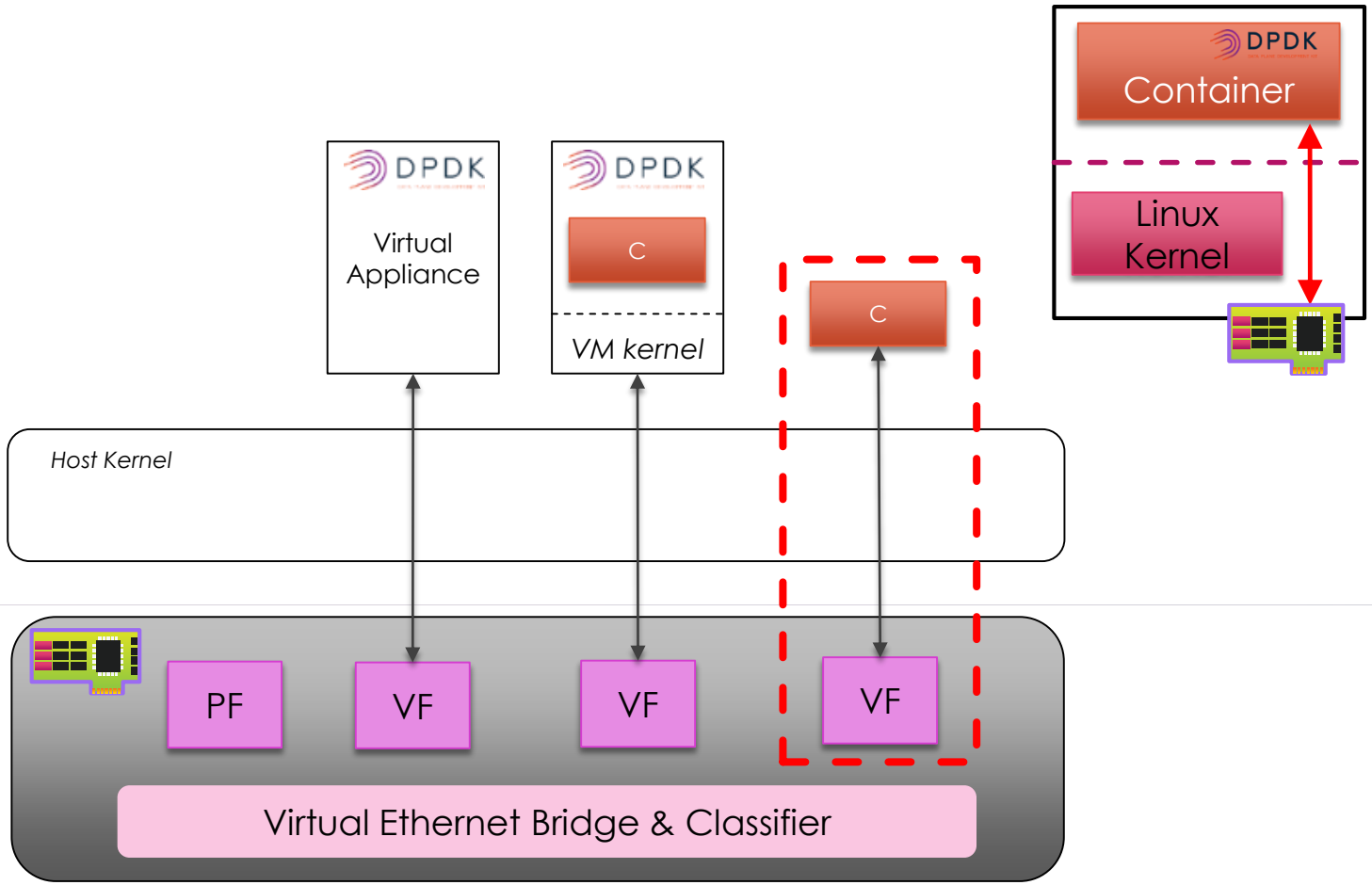
```
$ pqos -e "llc:2=0x00003"
$ pqos -a "llc:2=8,9,10"
```

CAT	Noisy Neighbor	DPDK IP Pipeline Application (Packet size = 64 Bytes, Flows = 16 Millions)	
		Throughput (Mpps)	LLC Occupancy (MB)
Not Present	Present	9.8	4.5
Present	Present	15	13.75

http://www.intel.com/content/www/us/en/communications/increasing-platform-determinism-pqos-dpdk-white-paper.html

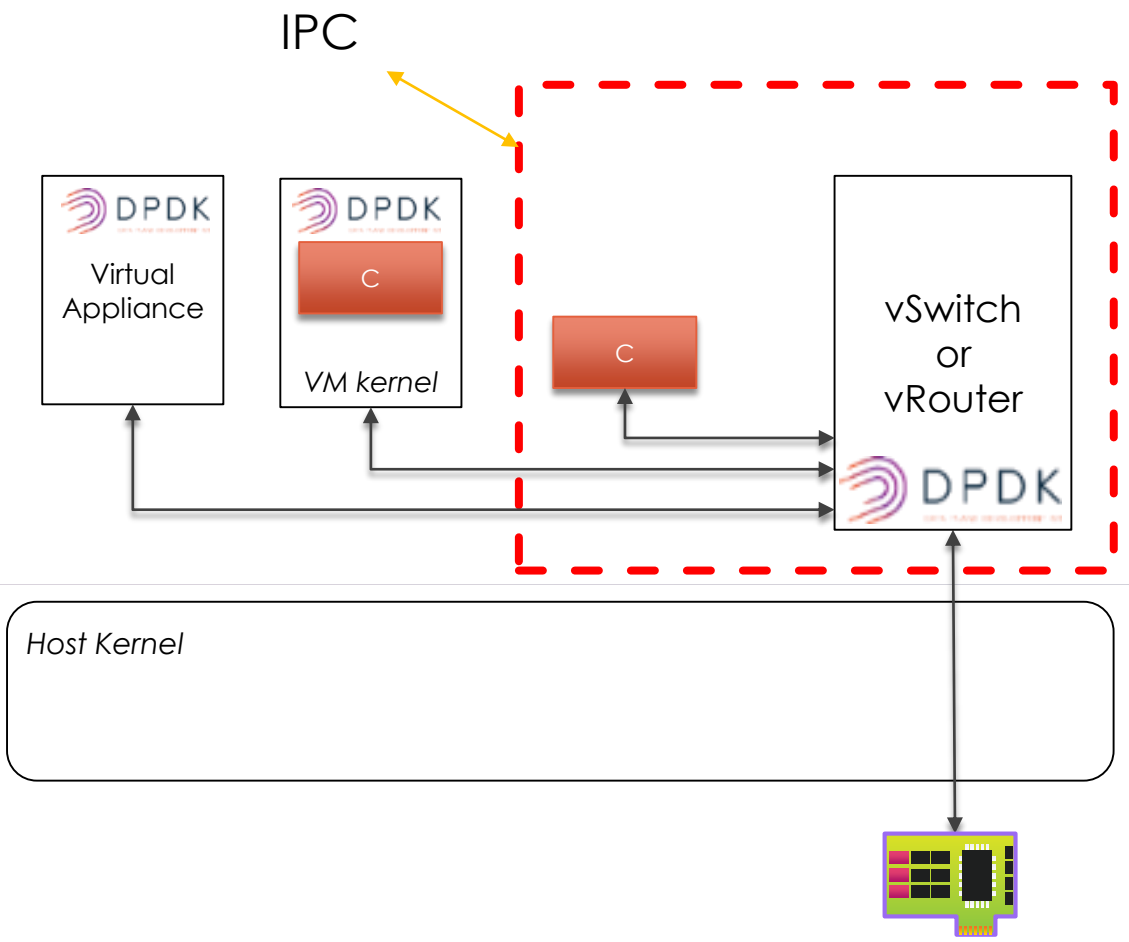
Userland SR-IOV in Container

- Pros:
  - Line rate even with small packets
  - Low latency
  - HW-based QoS
- Cons:
  - # of VFs is limited (64 or 128)
  - Not flexible (in need of router or switch with support of VTEP)



DPDK virtio_user for Container

- Problem statement from PV to IPC
- virtio ring as IPC, why?
  - Standard Protocol in Spec.
  - Consistent host backend
  - Performance
    - Bypass kernel
    - Share memory based
    - Smarter notification
    - Cache friendly
  - Security



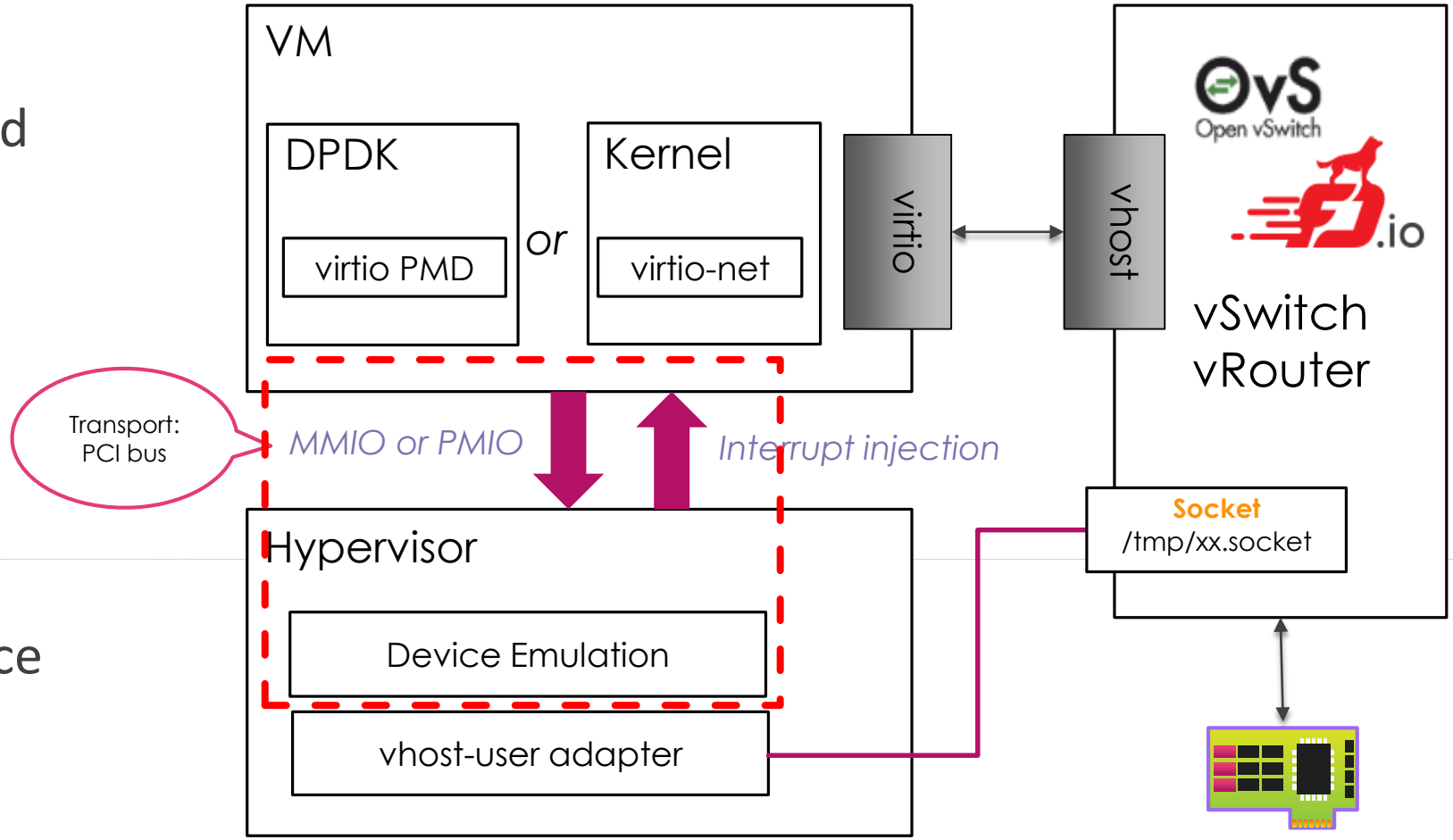
virtio approach for IPC

PV based

- virtio device is emulated in QEMU
- virtio can use various different bus (PCI Bus, MMIO, Channel I/O)

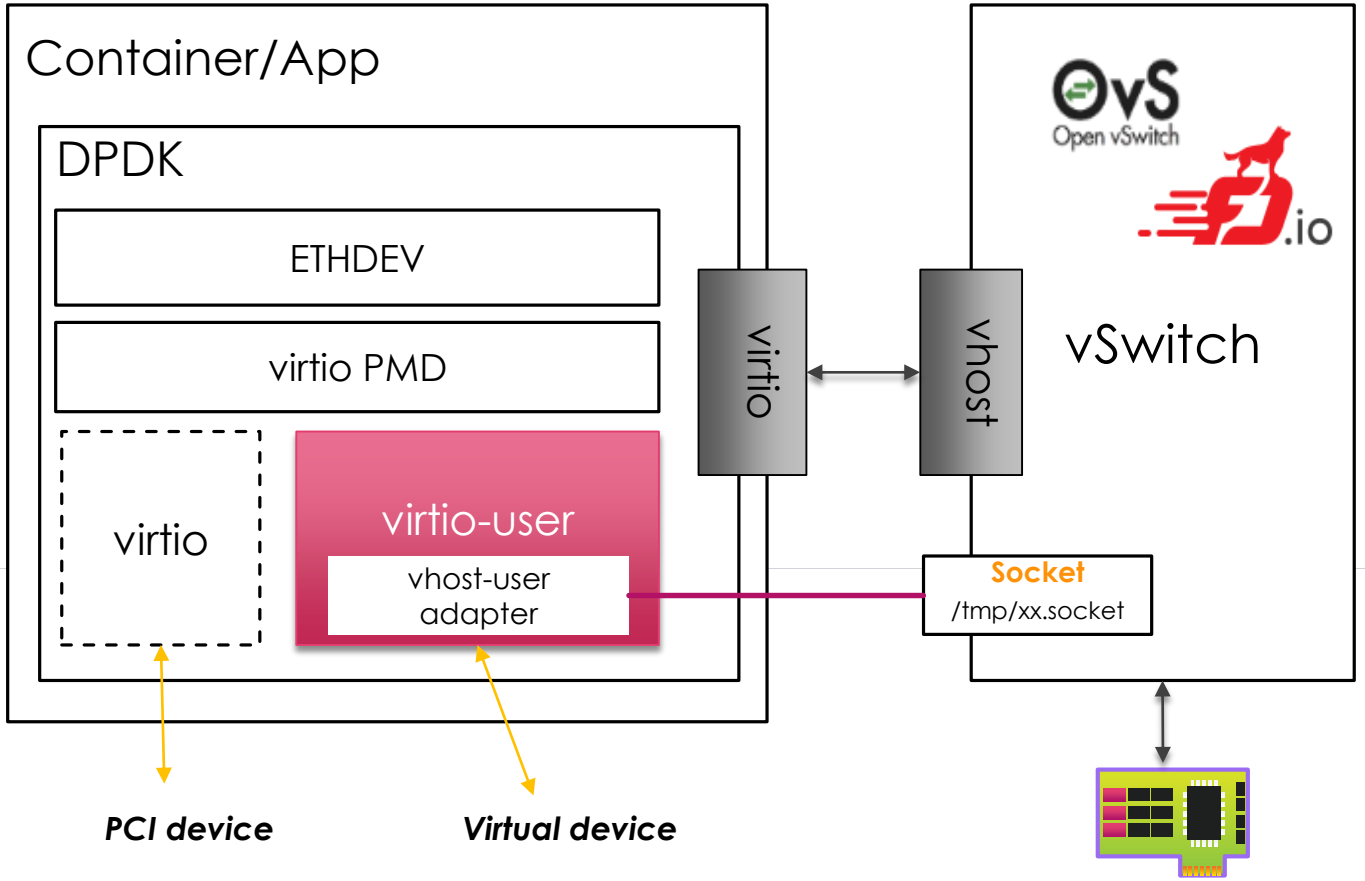
IPC based

- Transport bus and device emulation is no longer used.



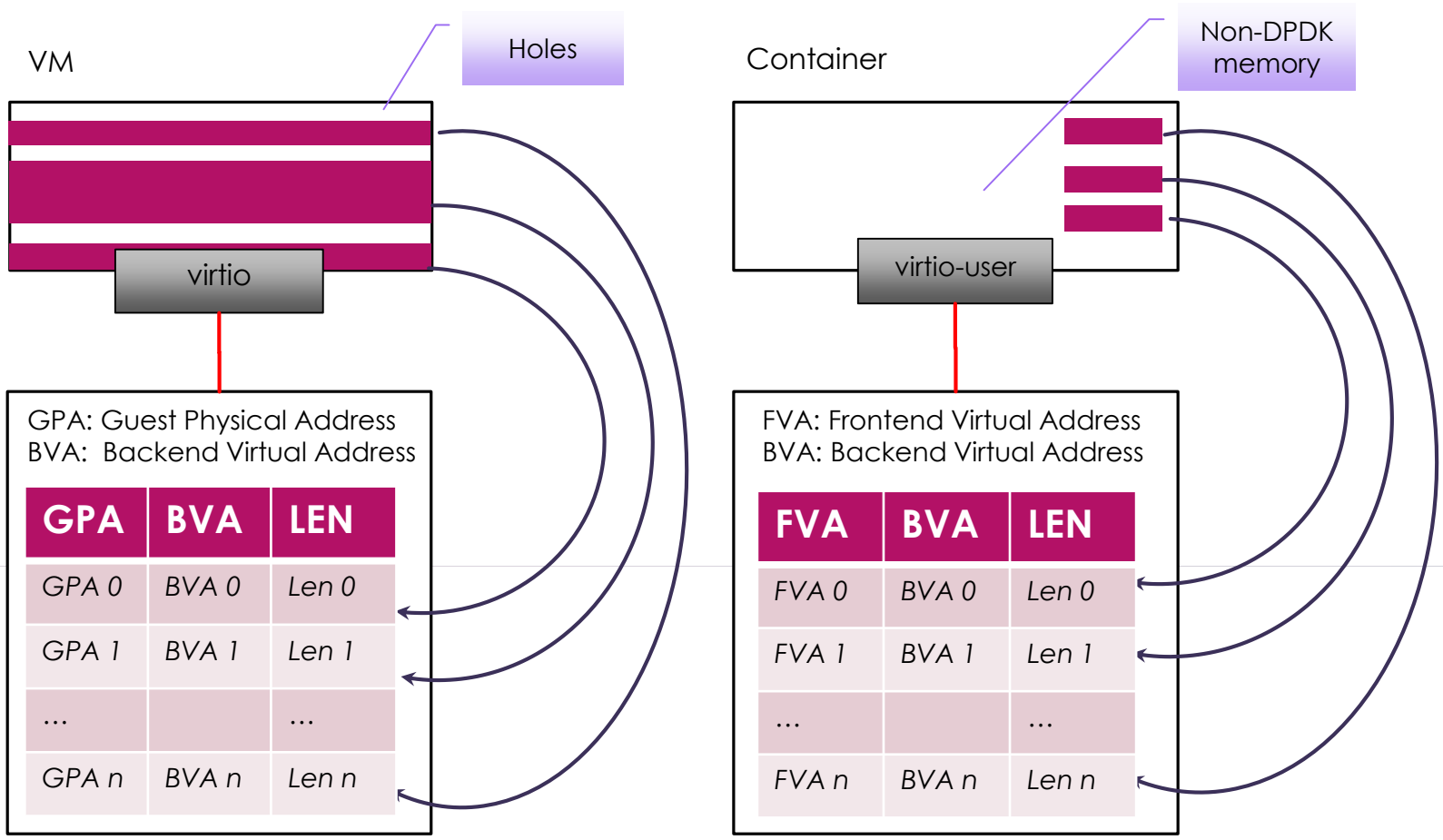
virtio_user intro (1)

- virtio-user as a DPDK virtual device (vdev)
- Talk to backend by vhost-user adapter w/o device emulation
- Single consistent vhost PMD on the backend



virtio_user intro (2)

- FVA to BVA address translation
- Memory mapping only for DPDK used memory
- Number of memory region is limited !



Setup virtio_user with OVS-DPDK

- Add a bridge and a vhost-user port in ovs-dpdk

```
$ ovs-vsctl add-br br0 -- set bridge br0 datapath_type=netdev
$ ovs-vsctl add-port br0 vhost-user-1 -- set Interface vhost-user-1
type=dpdkvhostuser
```

- Prepare hugetlbfs

```
$ mount -t hugetlbfs -o pagesize=2M,size=1024M none /mnt/huge_c0/
```

- Run container

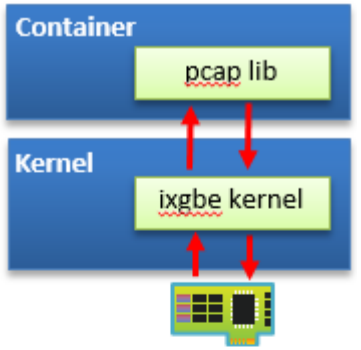
```
$ docker run … \
-v /usr/local/var/run/openvswitch/vhost-user-1:/var/run/usvhost \
-v /mnt/huge_c0/:/dev/hugepages/ \
…
-c 0x4 -n 4 --no-pci  --vdev=virtio-user0,path=/var/run/usvhost \
…
```

Performance Evaluation

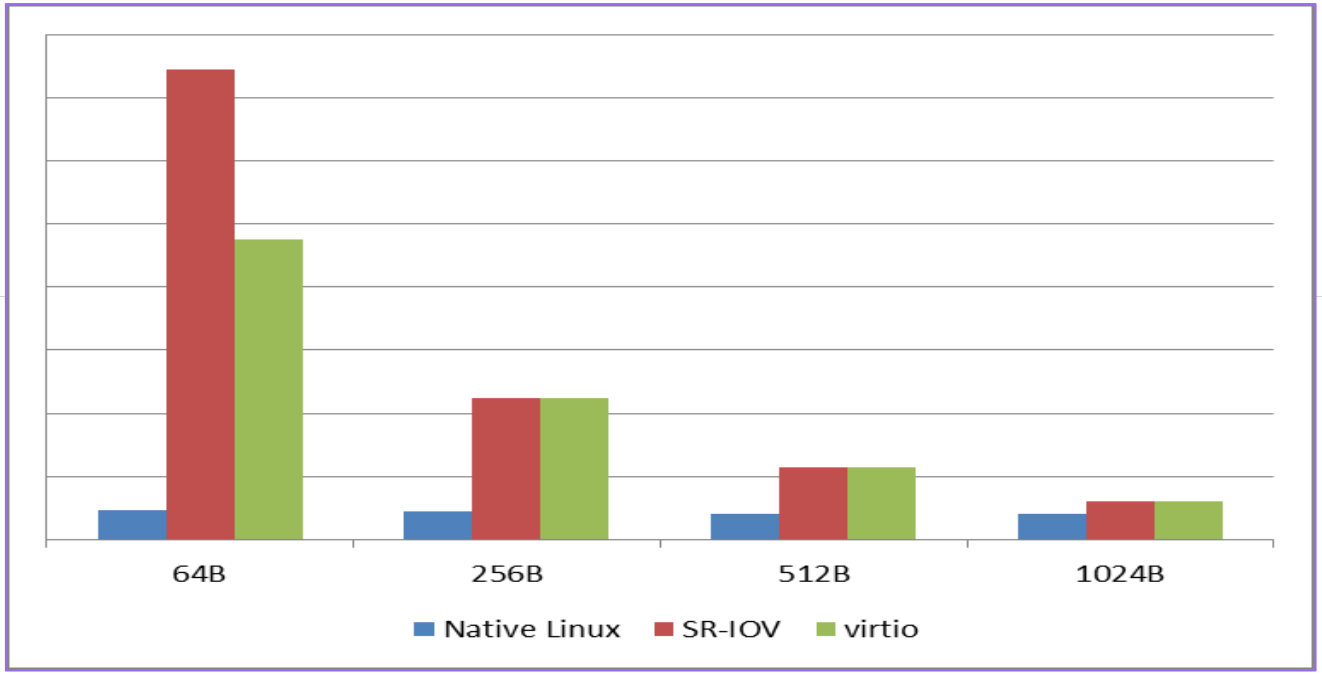
DPDK

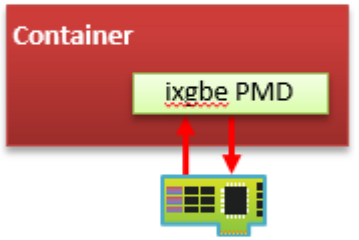
Latency

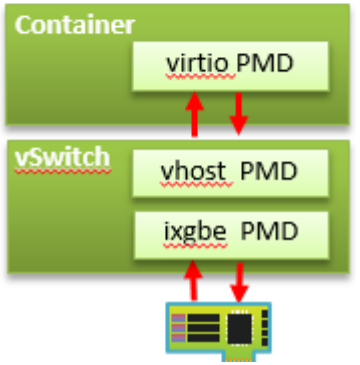
- For native Linux, ms level
- For the other two, us level
  - Polling mode, Batching, SIMD



Throughput







DPDK efforts Towards Container

- Hugetlb initialization process
  - sysfs is not containerized, and DPDK allocates all free pages
    - Addressed by here, avoid to use *-m* or *--socket-mem*
- Cores initialization
  - When/how to specify cores for DPDK?
    - Addressed by here, avoid to use *-c* or *-l* or *--lcores*
- Reduce boot time
  - Addressed by here and here

Run DPDK in Container Securely

- One container a hugetlbfs
- Run without --privileged
  - Run with --privileged is not secure
  - Larger attack face by leveraging NIC DMA
  - Why DPDK needs this? virt-to-phy translation
  - How to address? (see here)
    - Virtio does not need physical address
    - VF uses virtual address as the IOVA for IOMMU

Future work

- Single mem-backed file
- DPDK support container legacy network interface
- Interrupt mode of virtio (to scale)
- Long path to handle VF interrupts in userland (low latency)
- Integrate with popular orchestrators

Summary

- Use DPDK to accelerate container networking
  - Userland SR-IOV
  - Userland virtio_user (available in DPDK 16.07)
- Compared to traditional ways, it provides
  - High throughput
  - Low latency
  - Deterministic networking

Legal Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. **No computer system can be absolutely secure**. Check with your system manufacturer or retailer or learn more at intel.com.

Questions?

Cunming Liang
cunming.liang@intel.com