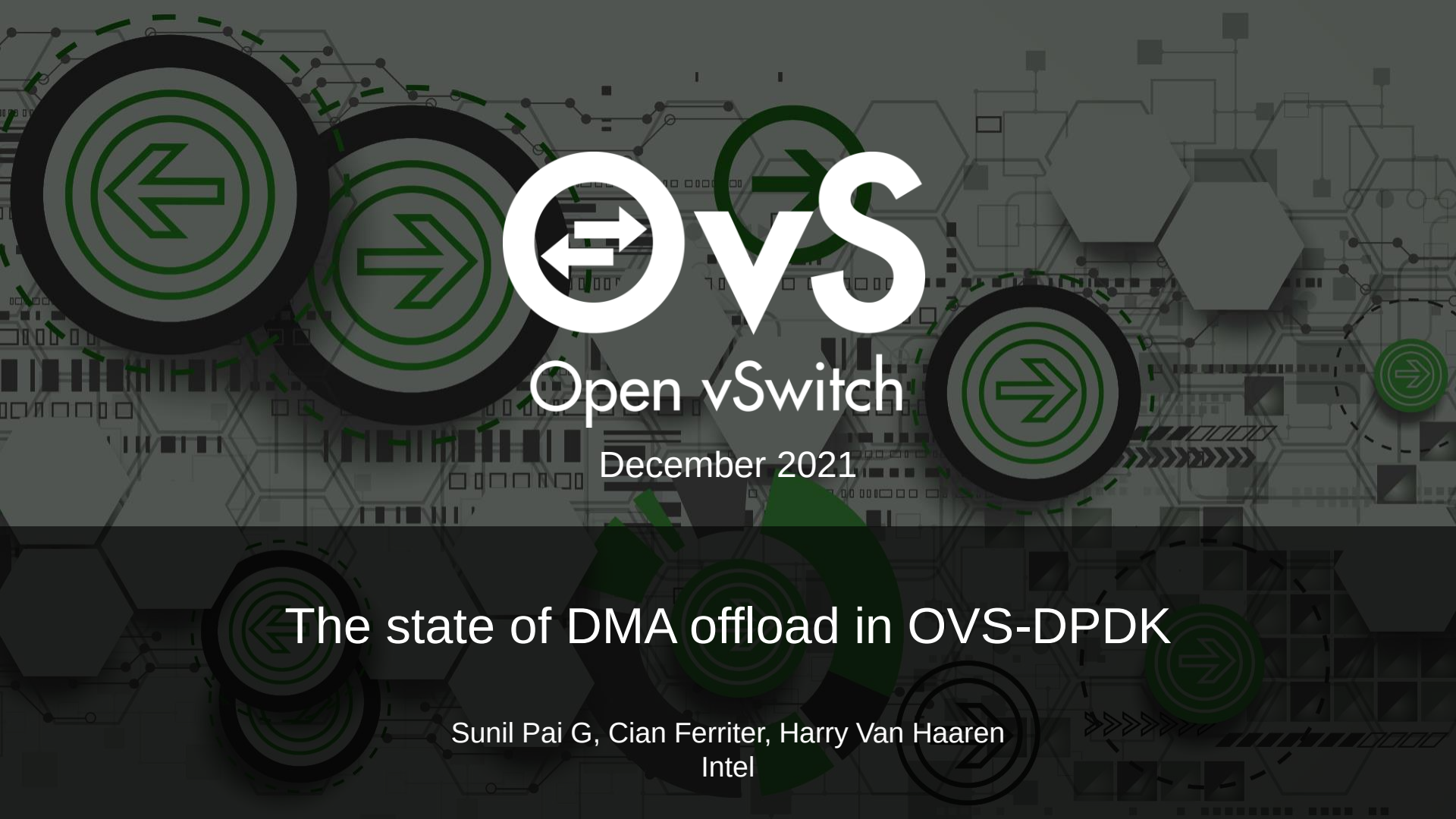


December 2021

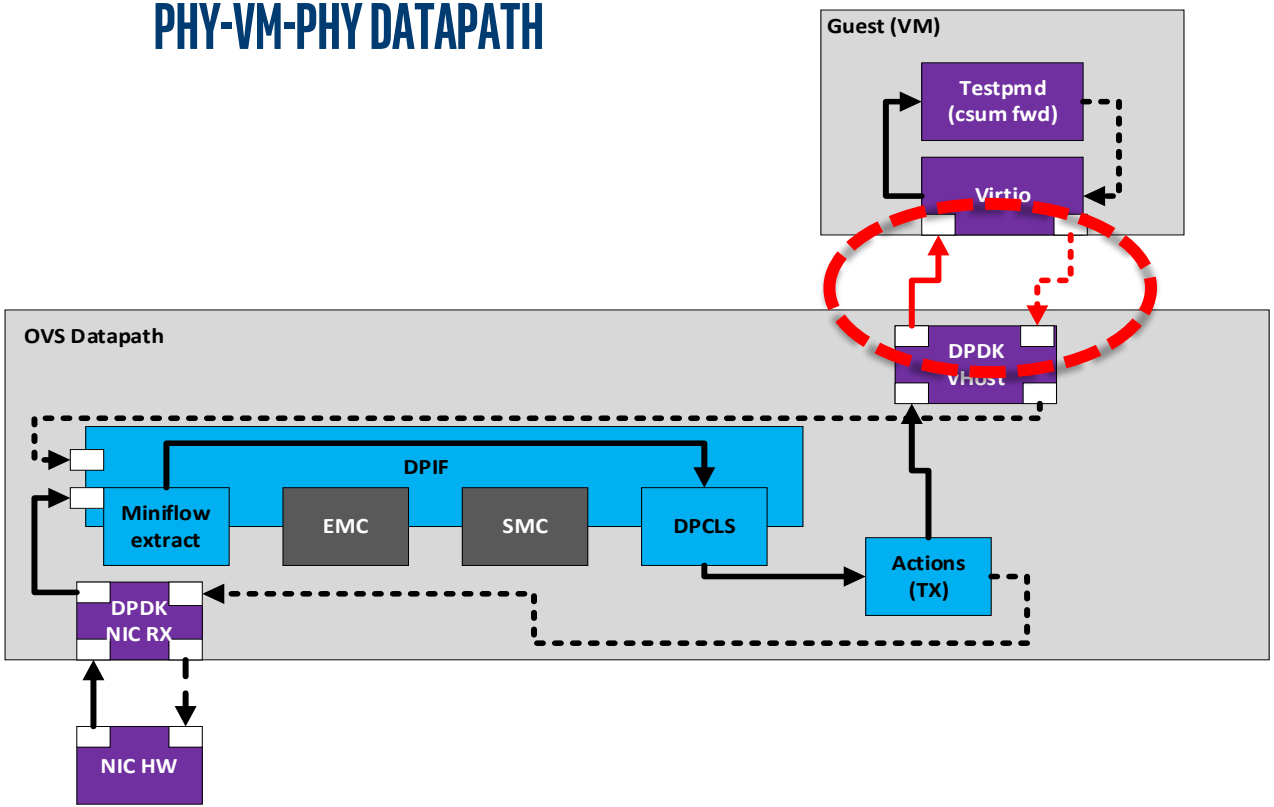
The state of DMA offload in OVS-DPDK

Sunil Pai G, Cian Ferriter, Harry Van Haaren
Intel

OVERVIEW

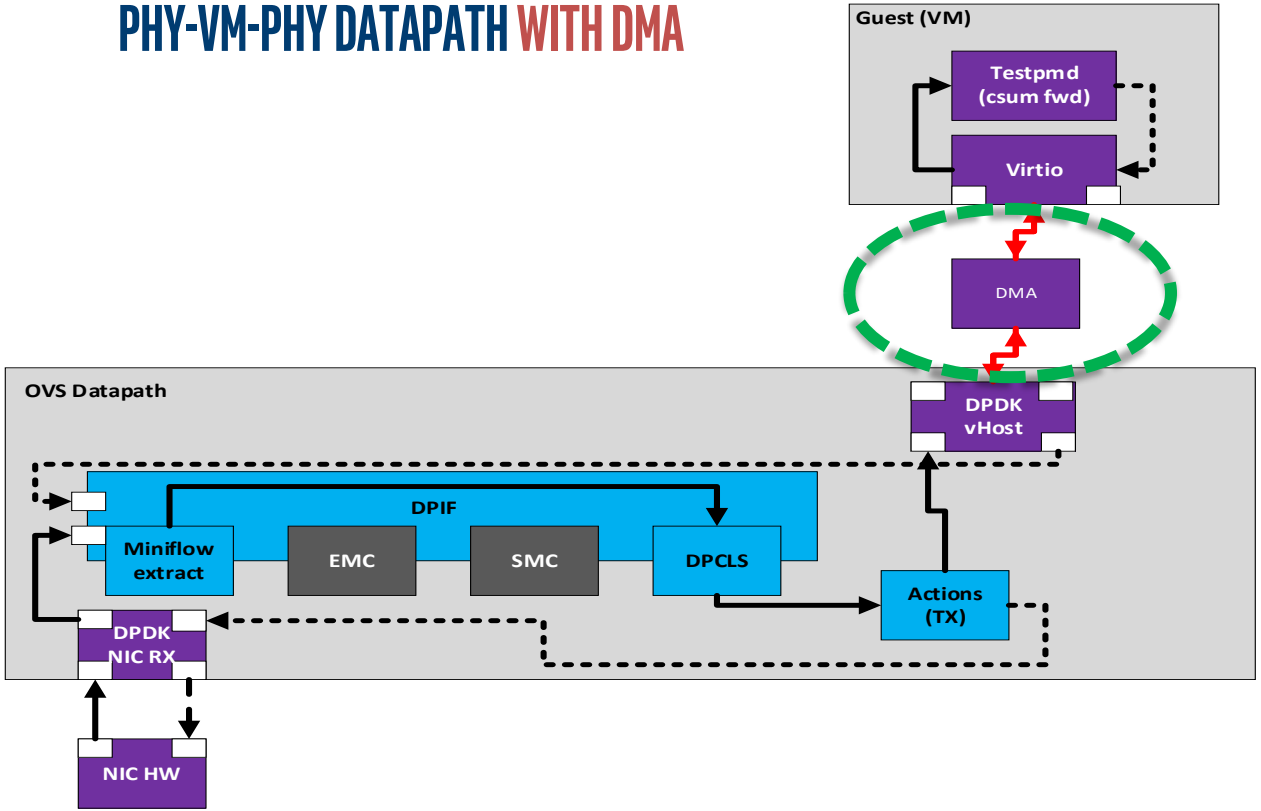
- The Problem
- The Solution
- Datapath design
- Scalability
- Packet walkthrough
- Future Work

The problem…

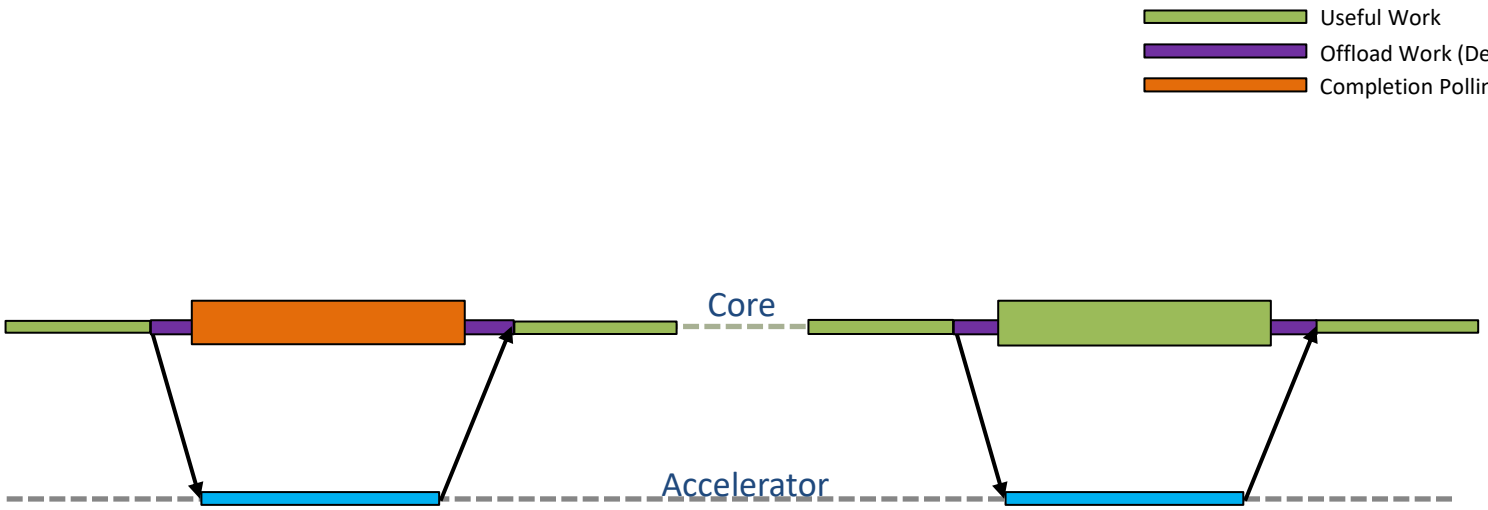

PHY-VM-PHY DATAPATH
Guest (VM)
Testpmd (csum fwd)
Virtio
DPDK vHost
OVS Datapath
DPIF
Miniflow extract
EMC
SMC
DPCLS
Actions (TX)
DPDK NIC RX
NIC HW

The solution...

PHY-VM-PHY DATAPATH WITH DMA



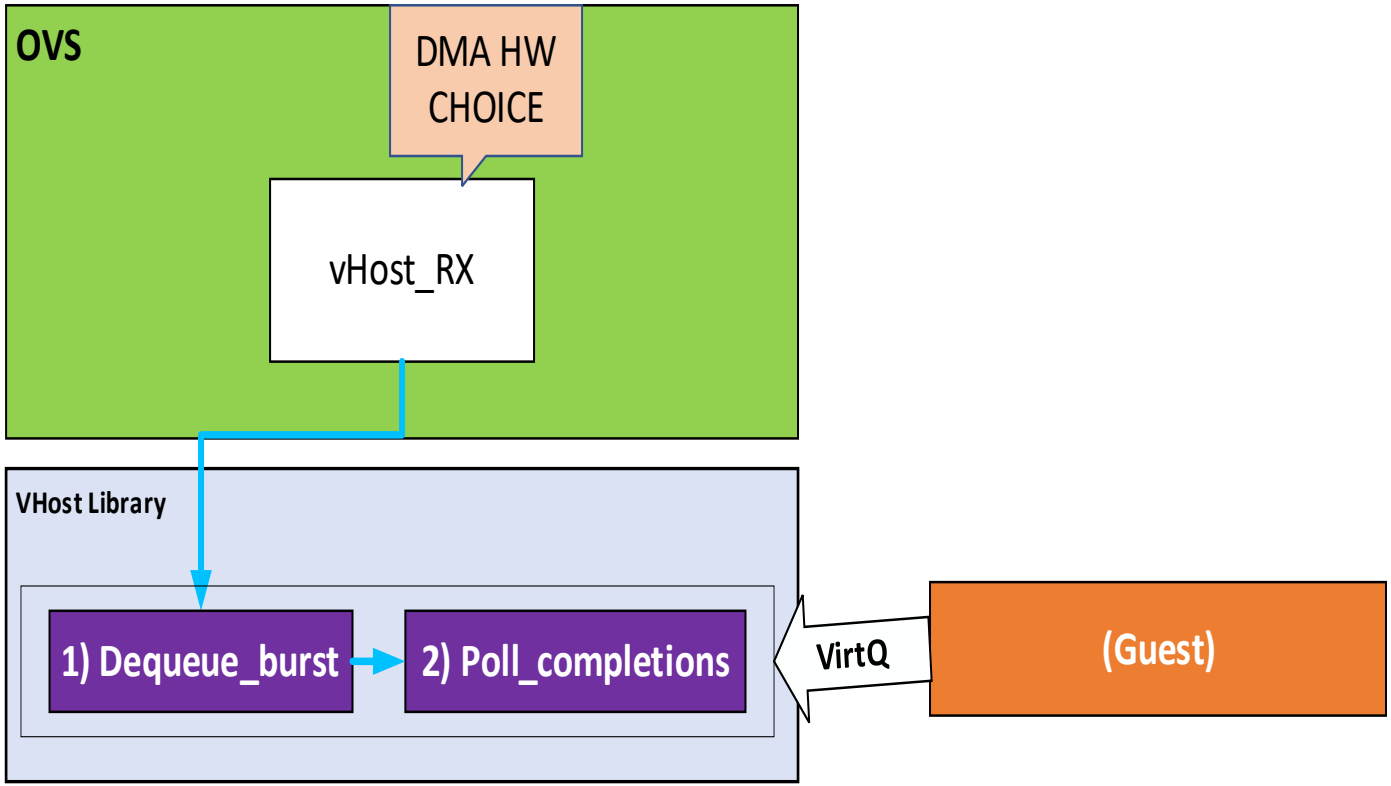
Offload types…



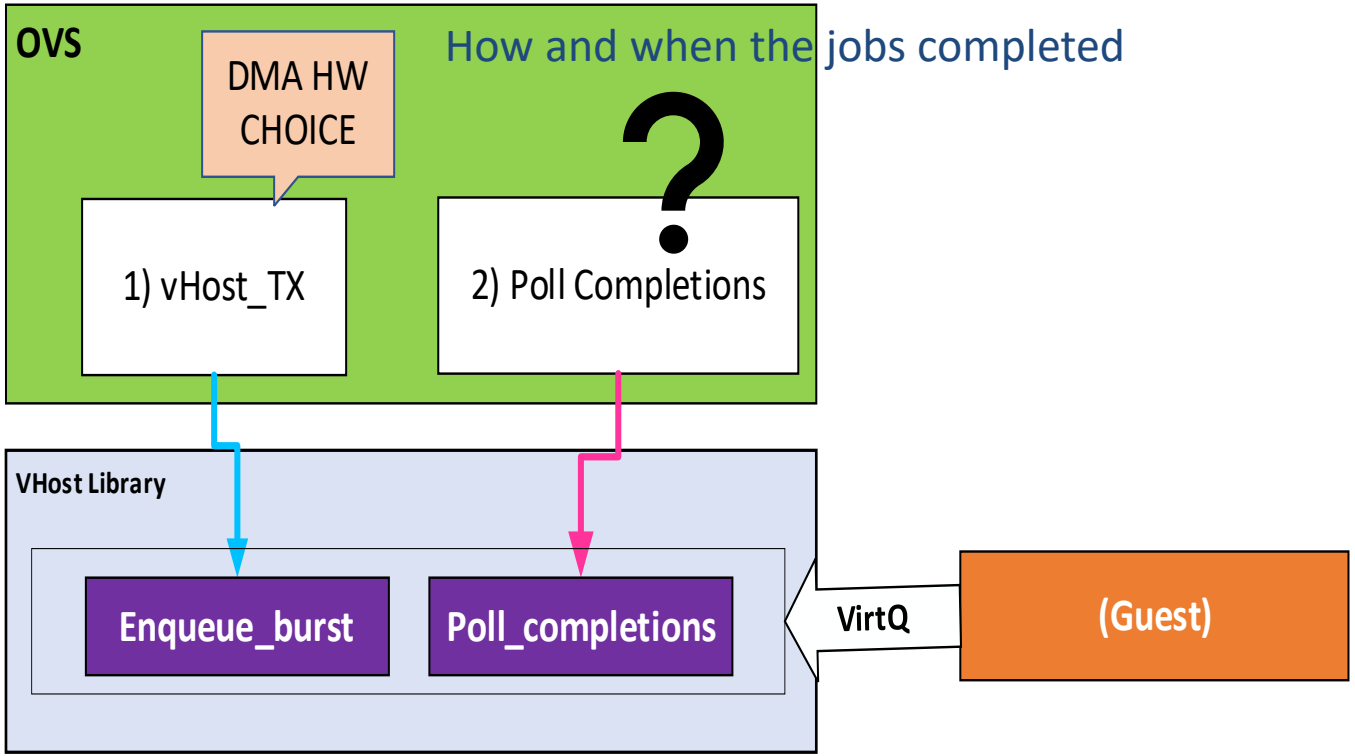
Synchronous Operation
– cycles wasted waiting for accelerator

Asynchronous Operation
– find other useful work to do while waiting on accelerator to avoid wasted cycles.

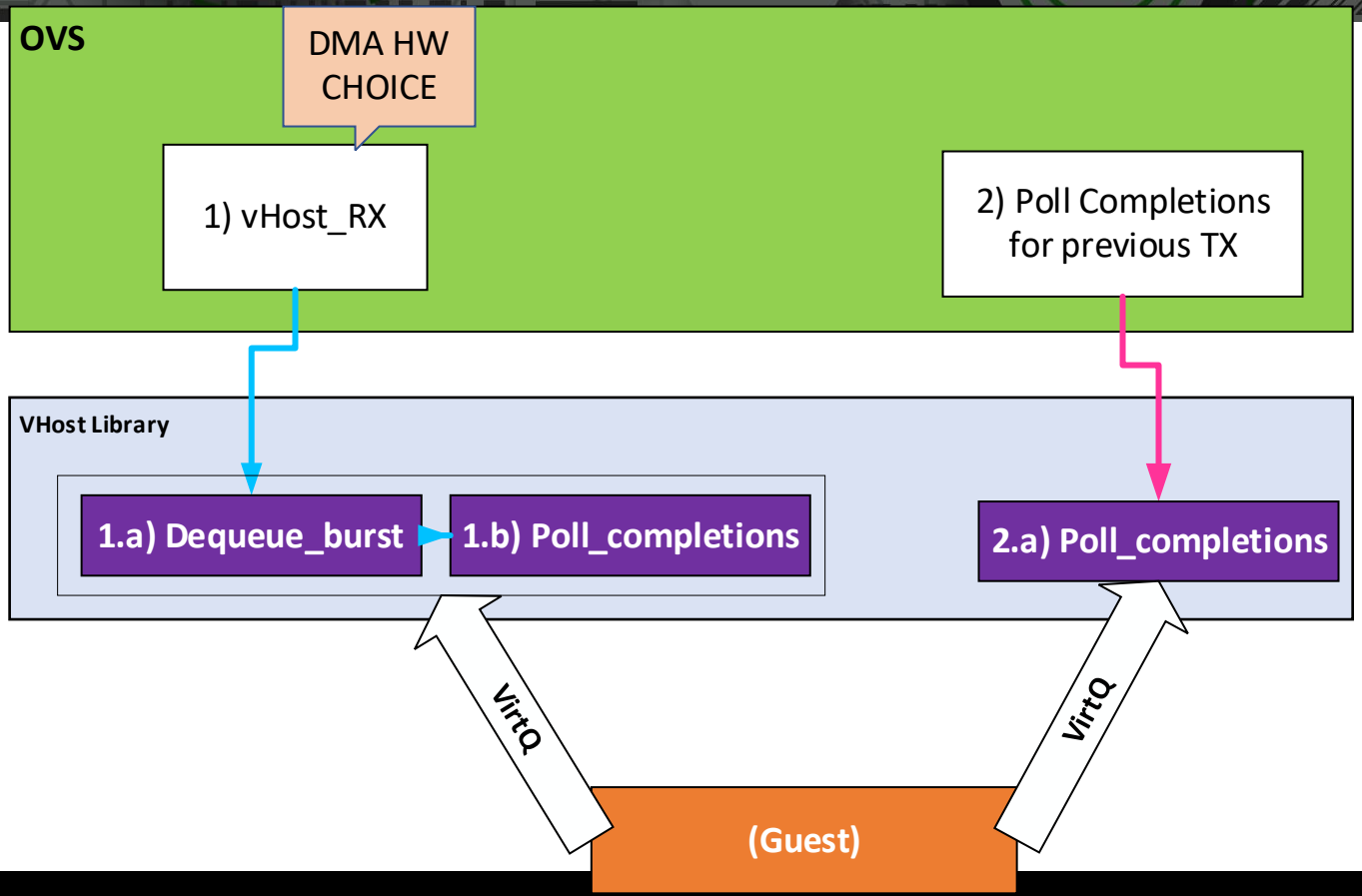
Datapath Design - Southbound



Datapath Design - Northbound



Datapath Design – Southbound (altered)



Scalability…

Mapping DMA to OVS?

Per Dataplane Thread

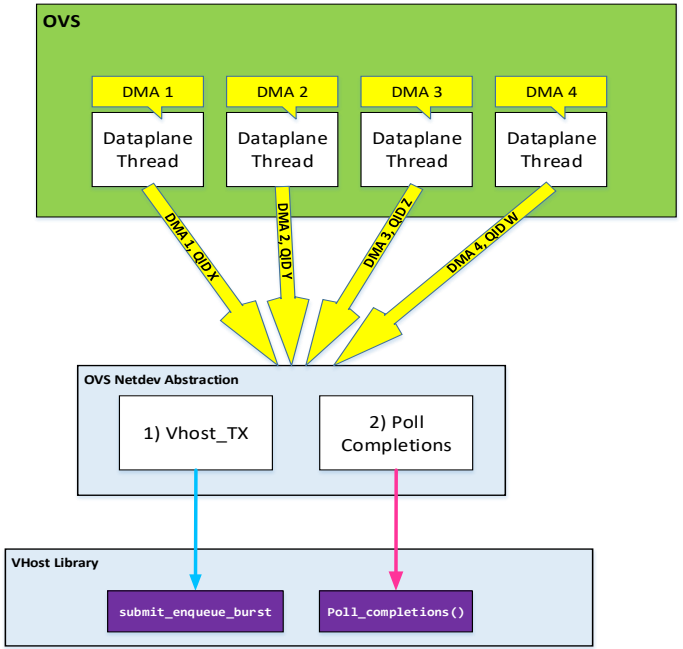
= Scalability!

➡ Add PMD thread

=

➡ Add DMA resource.

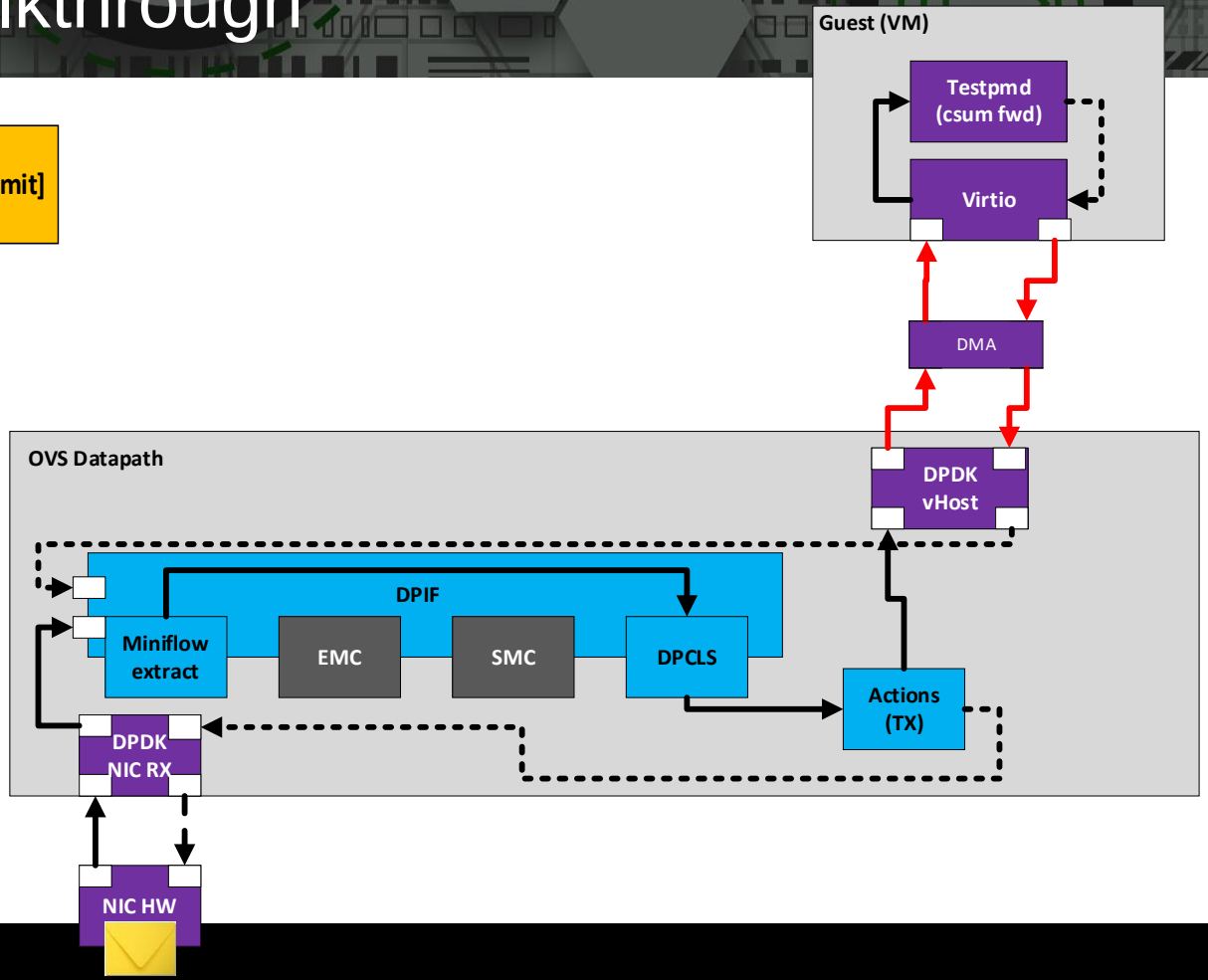
Balanced Switch/Copy performance.



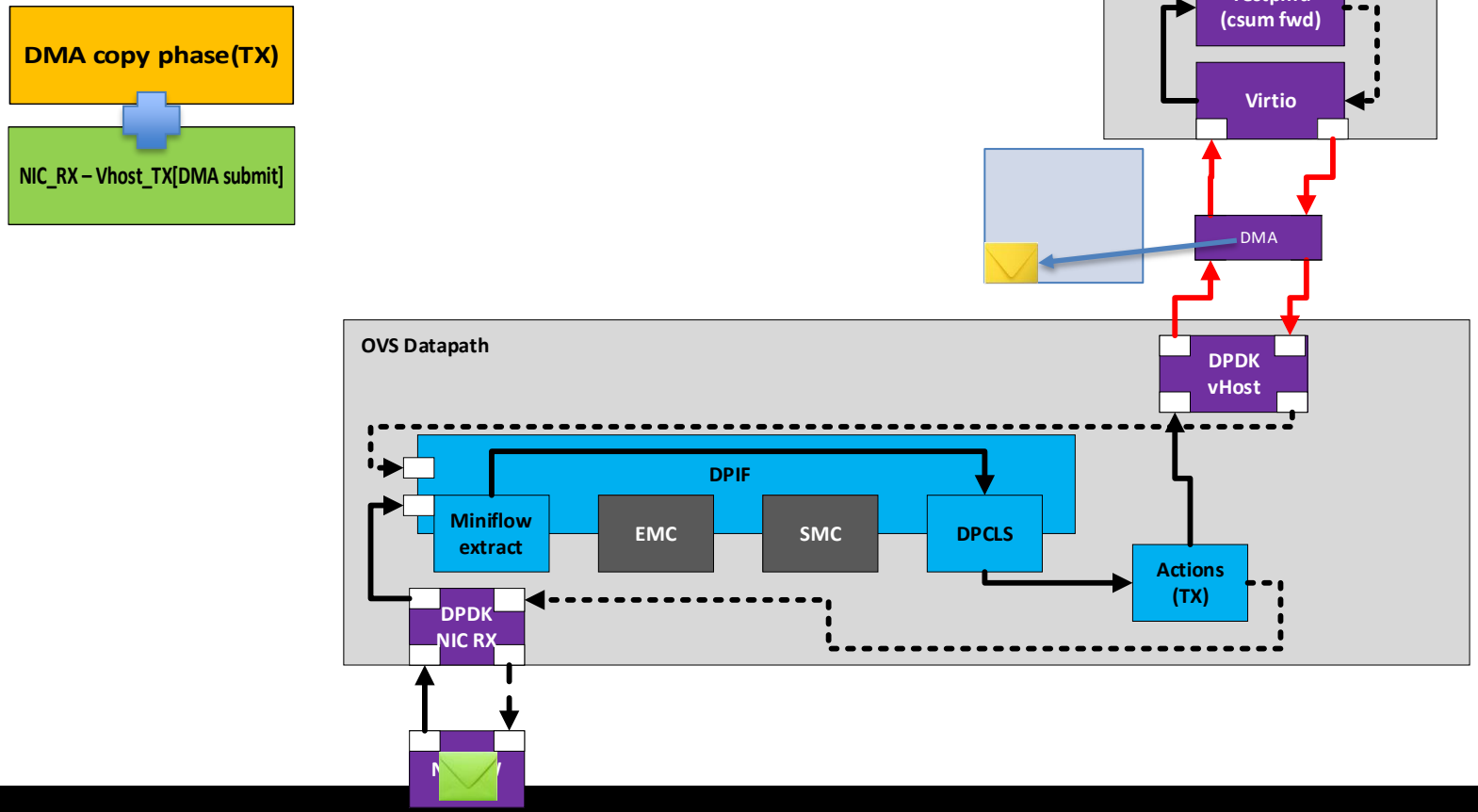
https://patches.dpdk.org/project/dpdk/patch/20211122105437.3534231-2-jiayu.hu@intel.com/

Packet walkthrough

NIC_RX – Vhost_TX [DMA submit]



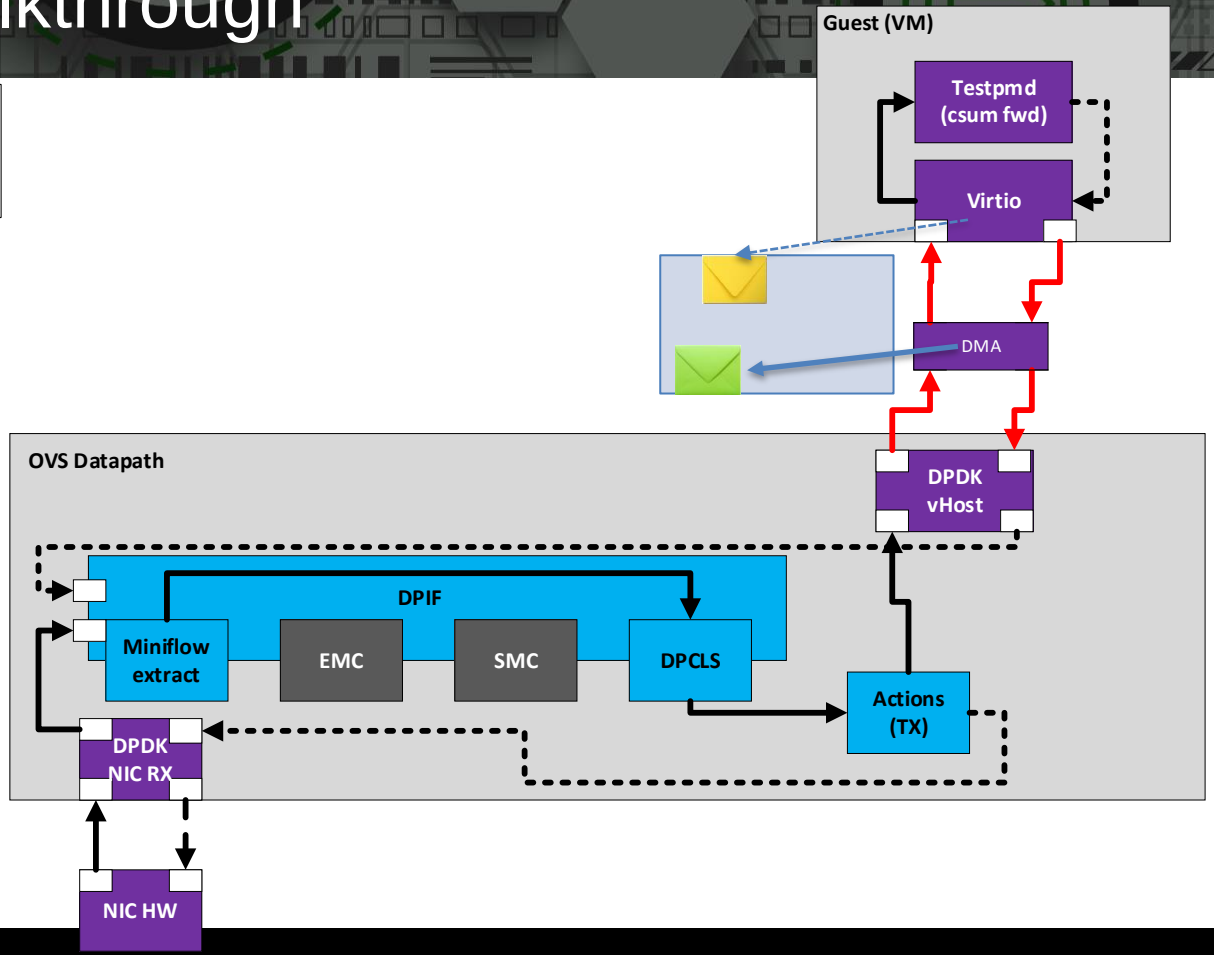
Packet walkthrough



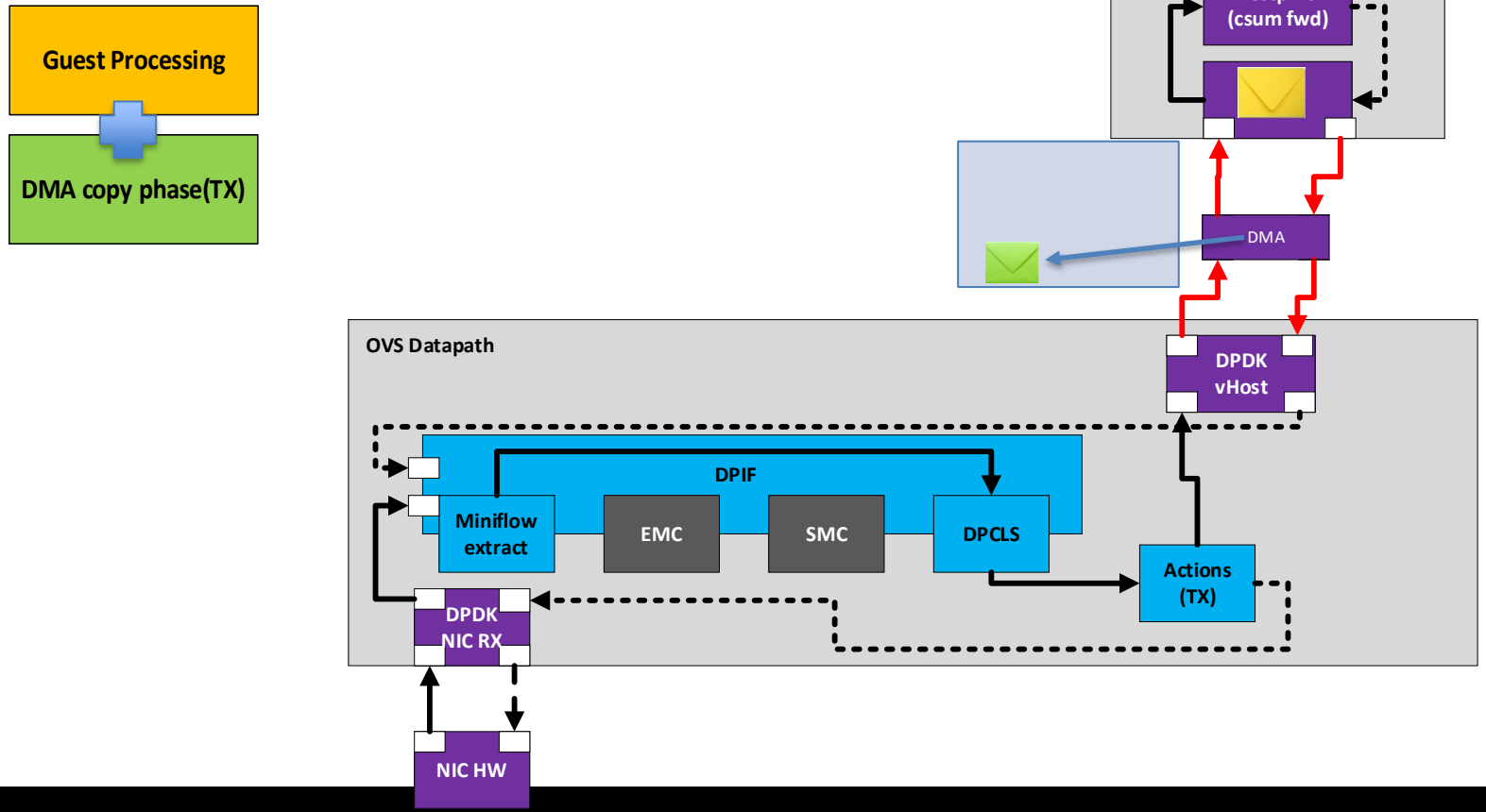
Packet walkthrough

Vhost_RX
[DMA submit] + [Poll completed]
+
Vhost_TX_Poll_completed

DMA copy phase(TX)

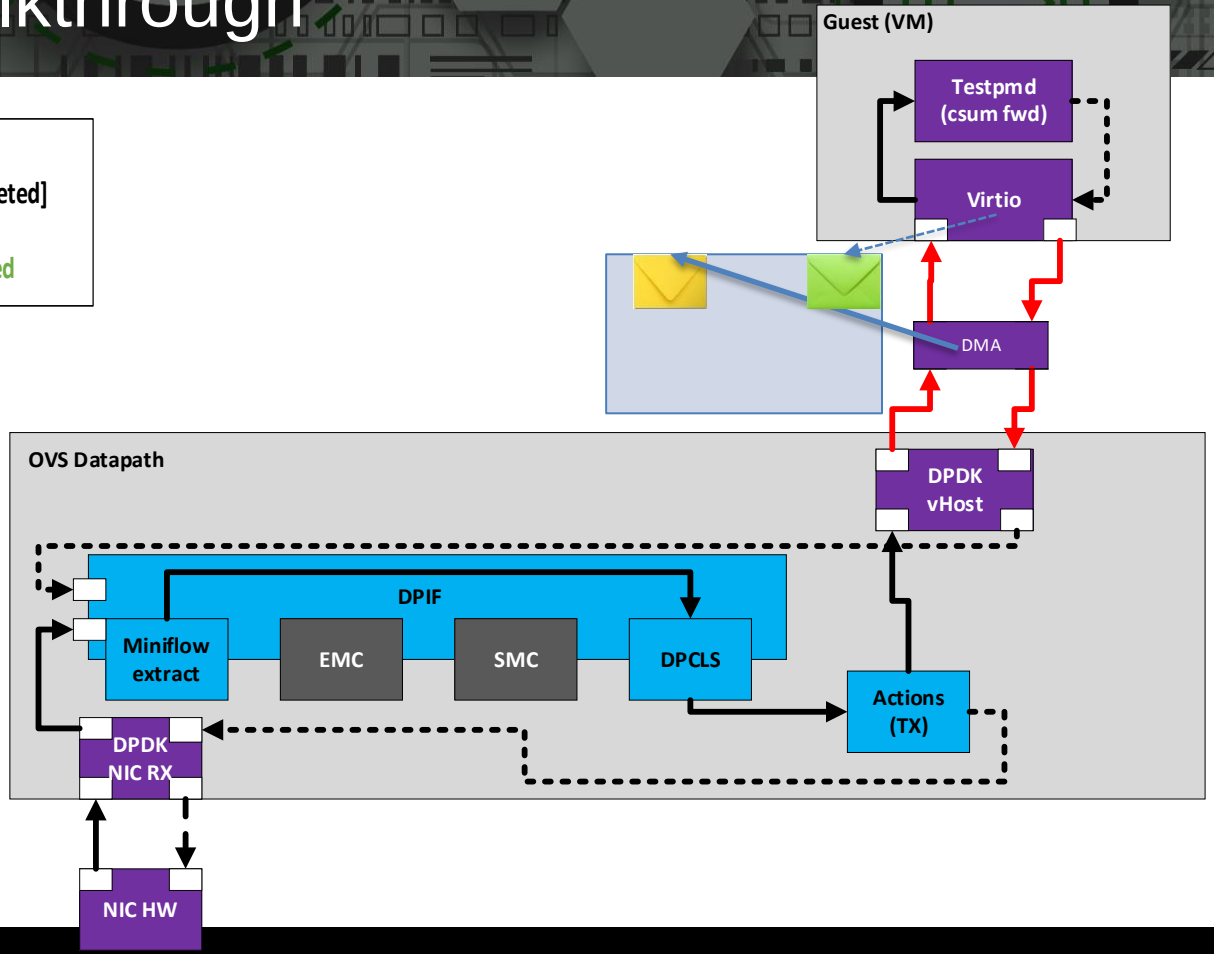


Packet walkthrough

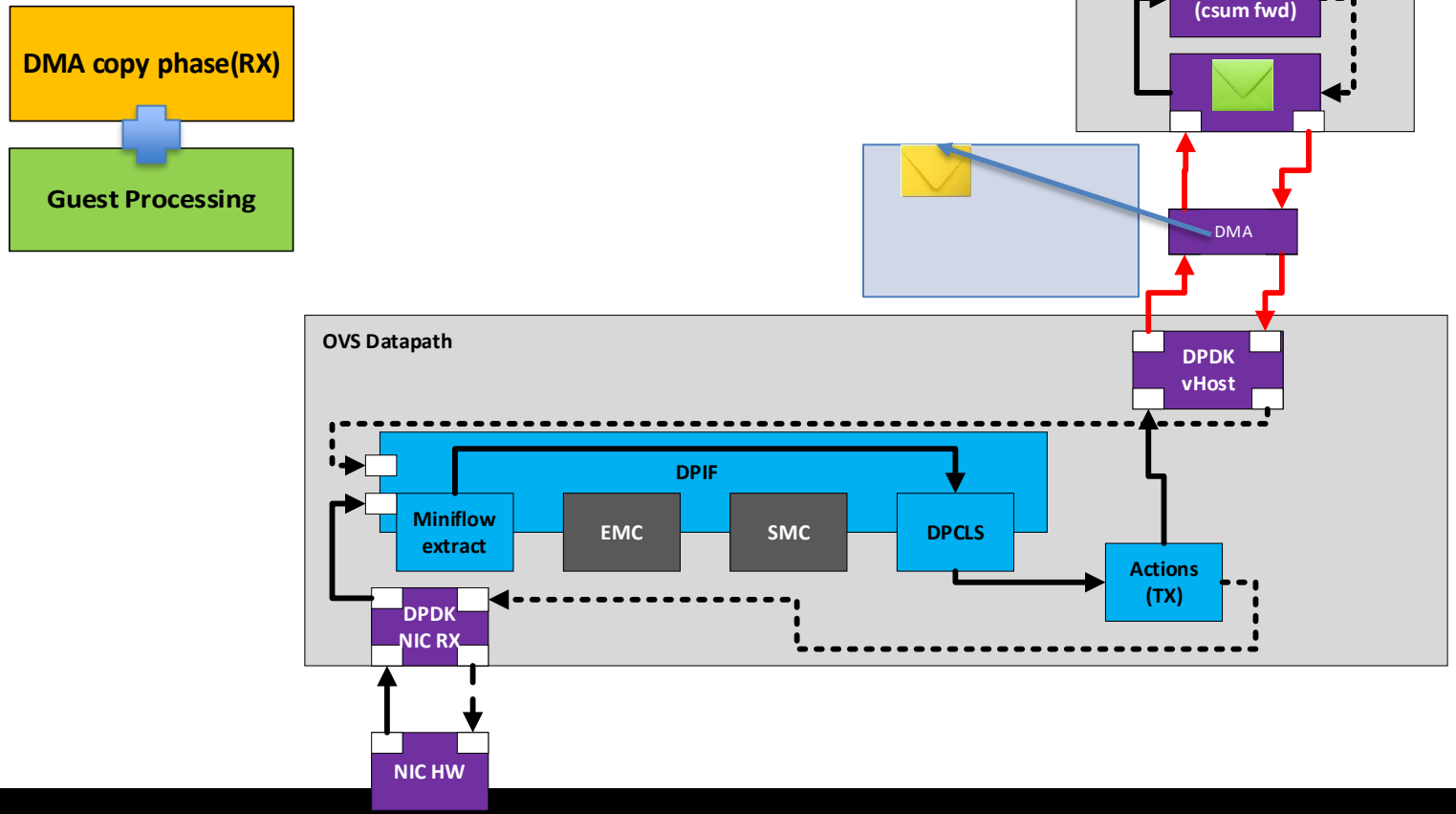


Packet walkthrough

Vhost_RX
[DMA submit] + [Poll completed]
+
Vhost_TX_Poll_completed

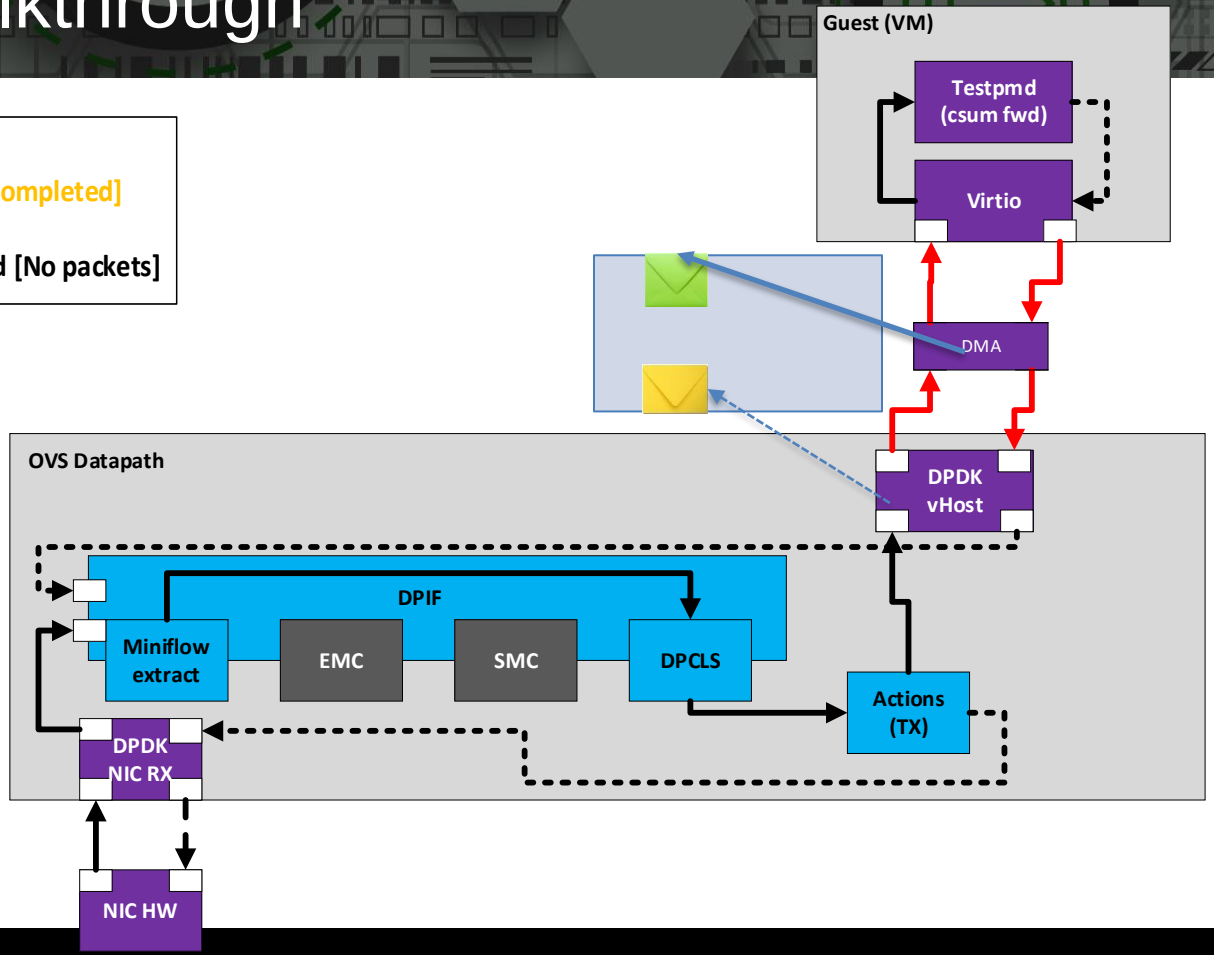


Packet walkthrough



Packet walkthrough

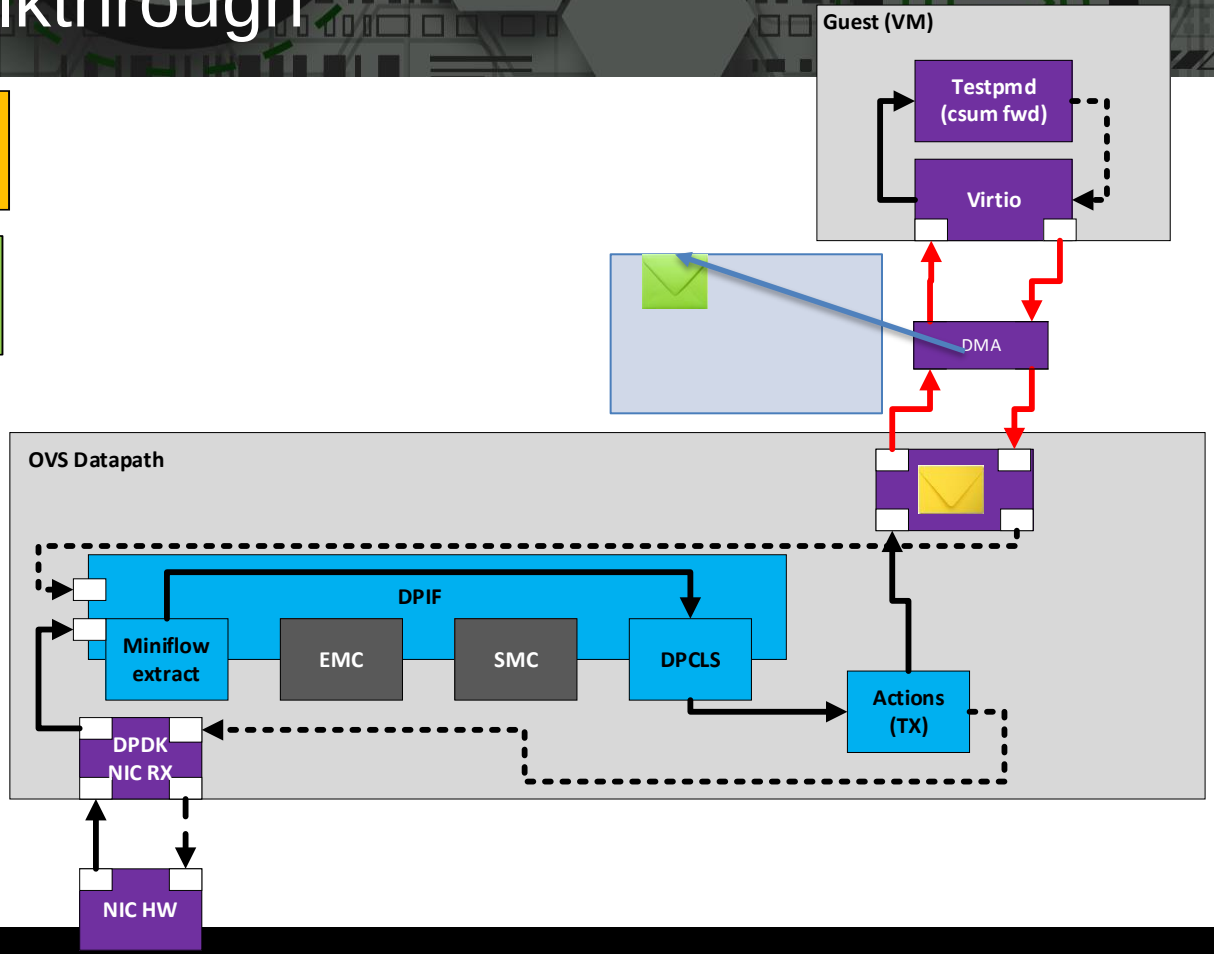
Vhost_RX
[DMA submit] + [Poll completed]
+
Vhost_TX_Poll_completed [No packets]



Packet walkthrough

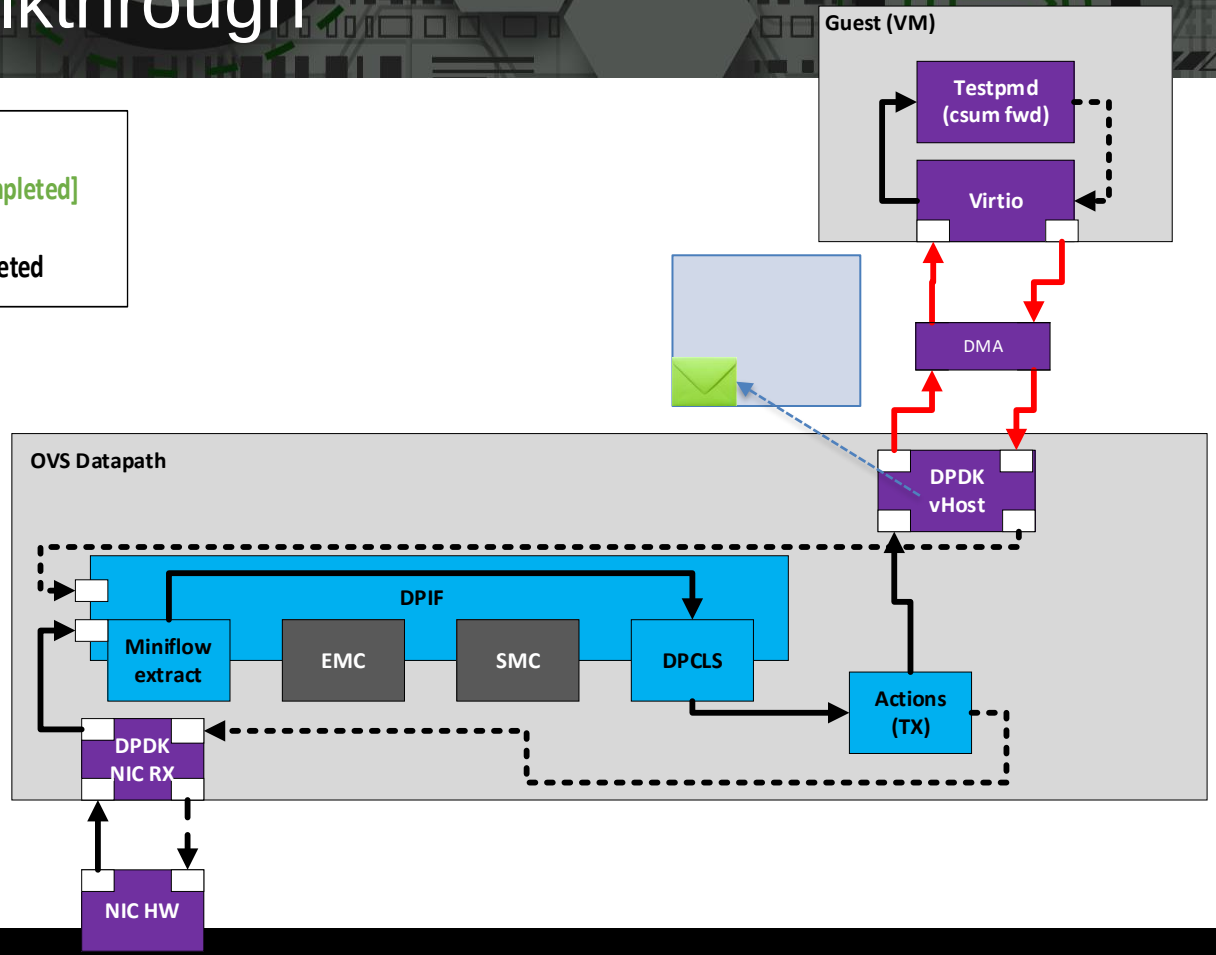
Vhost_RX – NIC_TX

DMA copy phase(RX)



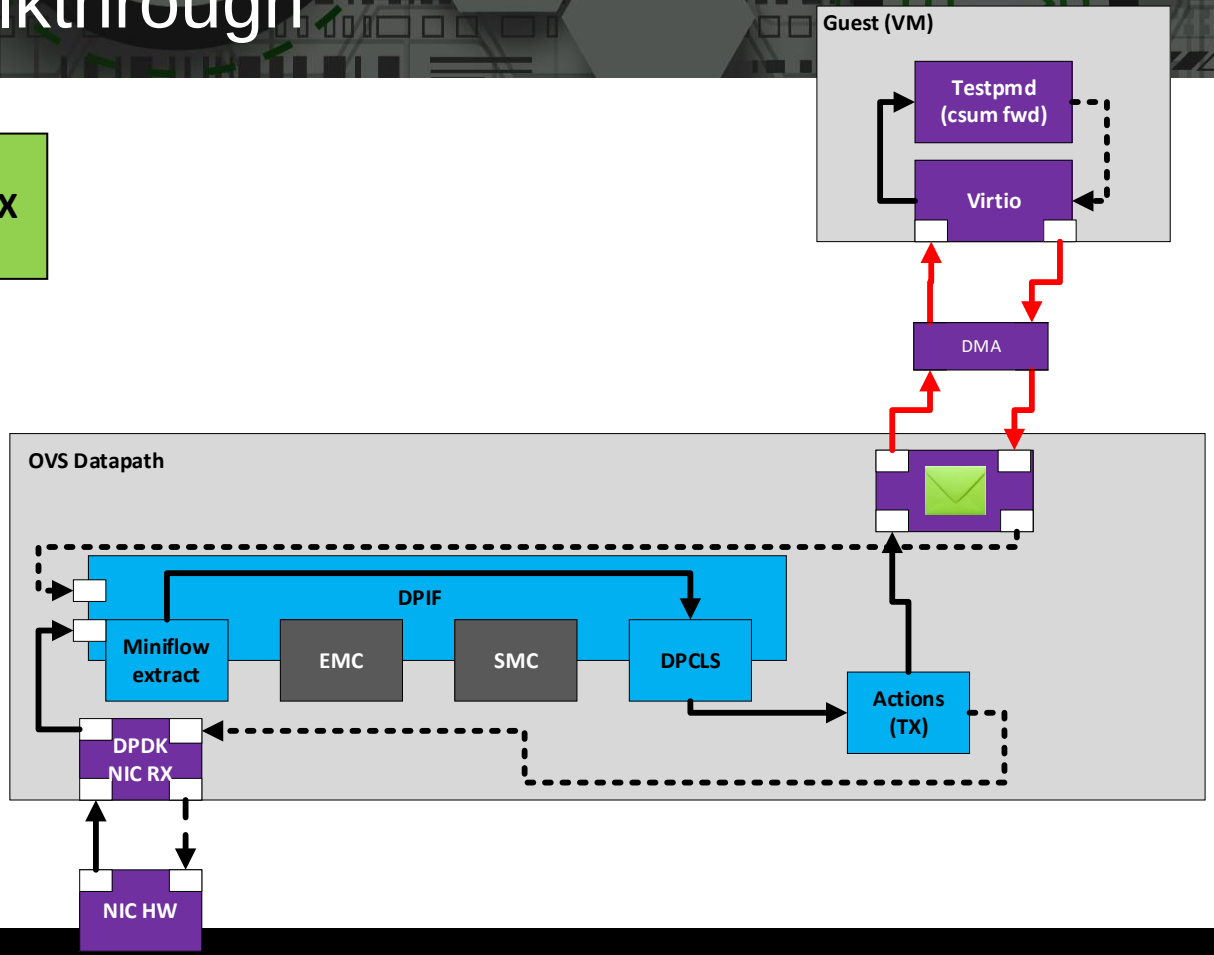
Packet walkthrough

Vhost_RX
[DMA submit] + [Poll completed]
+
Vhost_TX_Poll_completed



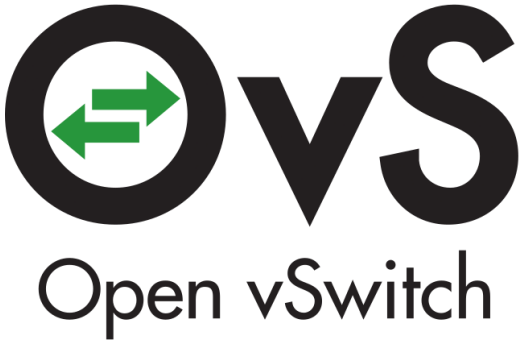
Packet walkthrough

Vhost_RX – NIC_TX



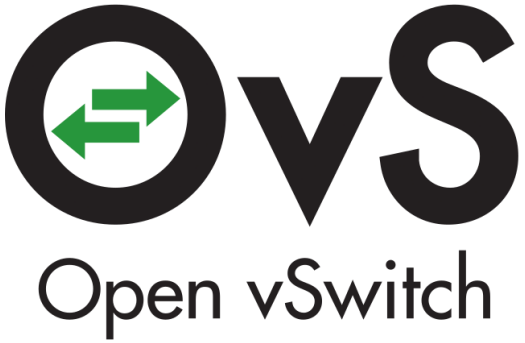
Next steps

- Upstream Design after community review



2.18

POC



2.19

UPSTREAM

! Thanks !
? Questions ?

Sunil Pai G
sunil.pai.g@intel.com

Cian Ferriter
cian.ferriter@intel.com

Harry van Haaren
harry.van.haaren@intel.com

References

Enabling asynchronous Para-virtual IO in OVS, OVS conference 2020

- Talk: https://www.youtube.com/watch?v=jj-0OQLe2oU
- Slides: https://www.openvswitch.org/support/ovscon2020/slides/Enabling-asynchronous-Para-virtual-IO-in-OVS.pdf

V2 RFC for vHost async implementation in OVS:

- http://patchwork.ozlabs.org/project/openvswitch/patch/20210907120021.4093604-2-sunil.pai.g@intel.com/

V1 RFC for vhost async implementation with DMAdev in vHost library in DPDK:

- https://patches.dpdk.org/project/dpdk/patch/20210823095355.2478423-2-sunil.pai.g@intel.com/
- https://patches.dpdk.org/project/dpdk/patch/20211122105437.3534231-2-jiayu.hu@intel.com/

Previous presentations in DPDK conferences:

- Asynchronous CBDMA Enqueue Framework for vHost-User, 2019

https://www.dpdk.org/wp-content/uploads/sites/35/2019/10/Asynchronous.pdf

- Accelerating Para-Virtual I/O with CBDMA, 2018

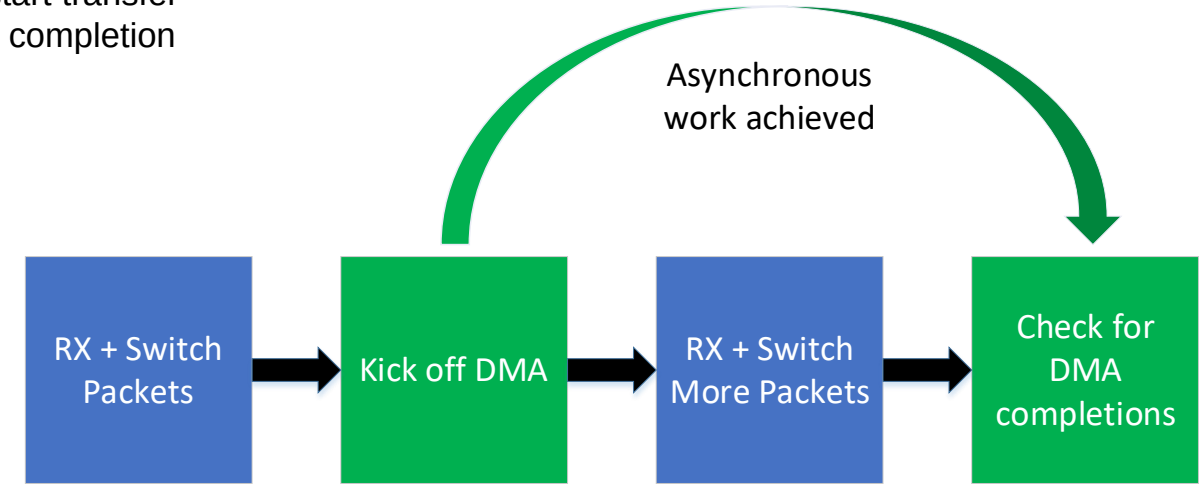
https://www.dpdk.org/wp-content/uploads/sites/35/2018/12/JiayuHu_Accelerating_paravirtio_with_CBDMA.pdf

BACKUP

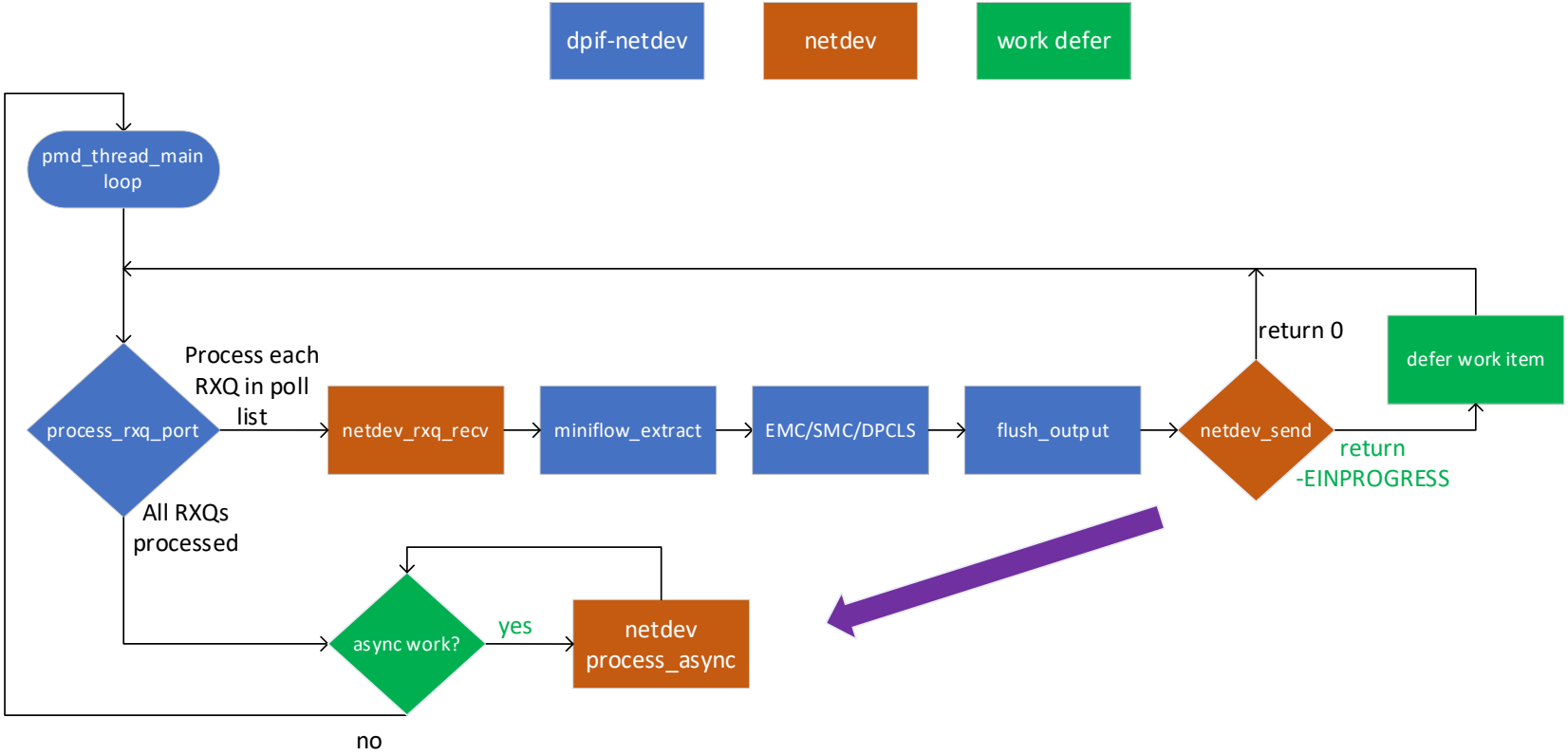
Why Defer of Work is Needed

- Packet copy into guest can be slow
  - Accelerate this using DMA Engine

- Don’t stall the CPU while the DMA Engine is active
  - Asynchronous acceleration requires defer of work
    - Call DPDK APIs to start transfer
    - Check back later for completion



Where Defer of Work Fits into OVS



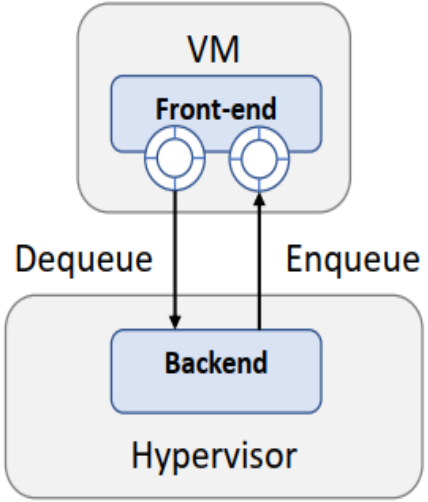
VirtIO

- Para-virtual I/O is a virtualization technique to enhance VM I/O performance.

- VirtIO is a standard of para-virtual I/O, which consists of **VirtIO front-end in VM** and **backend in hypervisor**.

- Back-end communicates with front-end by **copying packet buffers** between hypervisor's and VM's memory

- **Copying large bulk of data** between backend and frontend becomes a **hotspot**



DPDK API's

vHost async API's (vHost Library)

Northbound API's [Host to Guest]

- rte_vhost_submit_enqueue_burst
- rte_vhost_poll_enqueue_completed

Southbound API's [Guest to Host]

- rte_vhost_async_try_dequeue_burst

```
/* DMA callbacks */
struct rte_vhost_async_channel_ops {
        transfer_data(...);
        check_completed_copies(...);
};
```

- rte_vhost_async_channel_register_thread_unsafe
- rte_vhost_async_channel_unregister_thread_unsafe
- rte_vhost_async_get_inflight
- rte_vhost_clear_queue

- set RTE_VHOST_USER_ASYNC_COPY capability during rte_vhost_driver_register

DMAdev API's

Control path

- rte_dma_count_avail
- rte_dma_info_get
- rte_dma_configure
- rte_dma_vchan_setup
- rte_dma_start

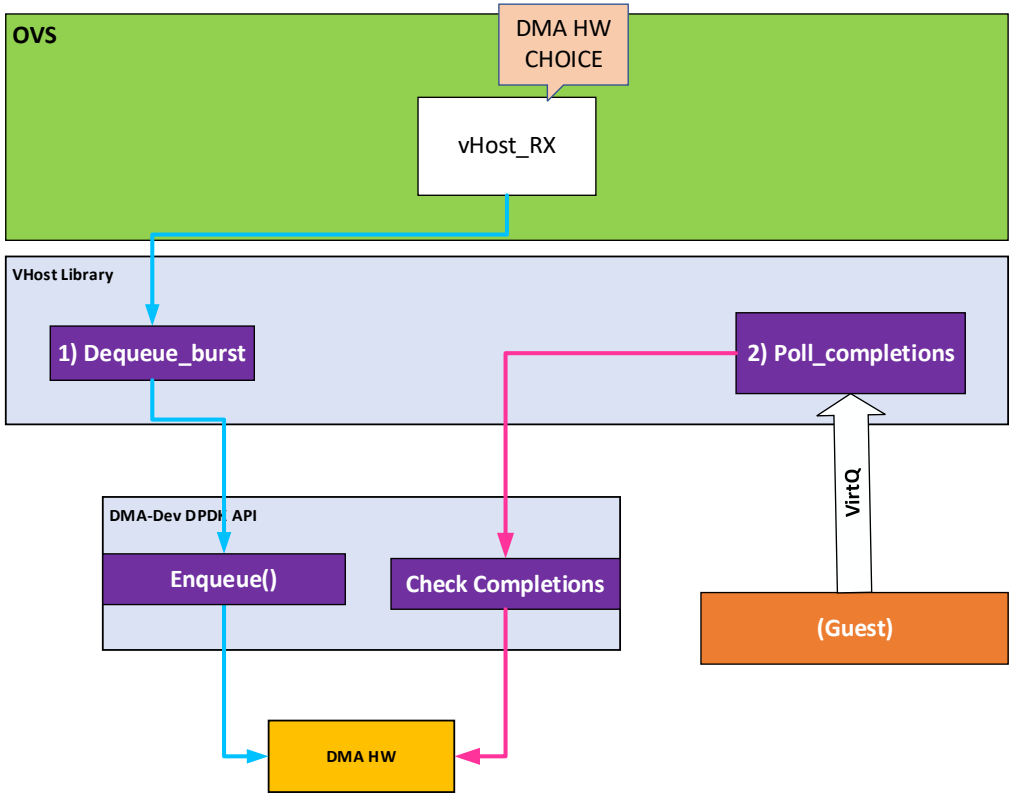
Datapath

- rte_dma_copy
- rte_dma_burst_capacity
- rte_dma_submit
- rte_dma_completed

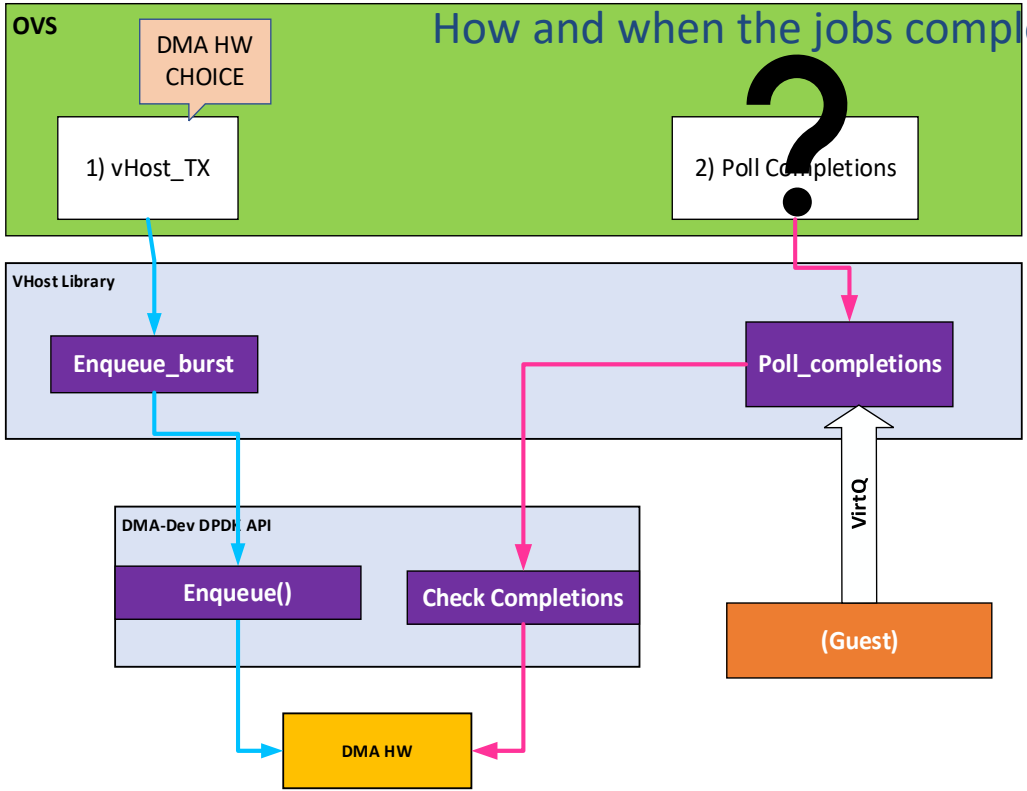
RFC based on above API set: http://patchwork.ozlabs.org/project/openvswitch/list/?series=261277

Note: **All vHost async and DMAdev API's are experimental**

Datapath Design - Southbound



Datapath Design - Northbound



Poll TX from RX context

Datapath Design – Southbound (altered)